VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Sami Pitkäkangas

AVOJOHTOVERKON LANGATON VAIHEVIRTOJEN MITTAUS

Tekniikka
2017

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Sami Pitkäkangas
Opinnäytetyön nimi	Avojohtoverkon langaton vaihevirtojen mittaus
Vuosi	2017
Kieli	suomi
Sivumäärä	59
Ohjaaja	Jukka Matila

Opinnäytetyö toteutettiin Vaspec Oy:lle. Työn tavoitteena oli kehittää sähköverkon avojohtolinjan langattomaan vaihevirtojen mittaamiseen uusi ratkaisu hyödyntäen LoRa-tekniikkaa. Ratkaisu tulisi sisältämään tiedonsiirtoon tarvittavan tekniikan avojohtopylväästä ala-asemaan.

Opinnäytetyössä tarkasteltiin kolmea mahdollista tekniikkaa langattoman tiedonsiirron toteuttamiseksi avojohtolinjalta ala-asemaan, joista yhtä langatonta tekniikkaa hyödyntäen kehitettiin mahdollinen ratkaisu langattoman tiedonsiirron toteuttamiseksi.

Työssä testilaitteistona toimi Linear Dust Networksn valmistama SmartMesh IP RF Certified Starter Kit, johon toteutettiin Windowspohjainen ohjainohjelmisto C-ohjelmointikielellä. Ohjelman tavoitteena oli saavuttaa toimiva kommunikaatio langattomaa tekniikkaa käyttäen vastaanottimelta keskusyksikölle.

Opinnäytetyössä saatiin kehitettyä lähes valmis ohjelmisto langattomaan tiedonsiirtoon, joka käyttää Lora-tekniikalla toimivaa laitteistoa. Tämän opinnäytetyön ohjelmistoa ja muita osa-alueita voitaisiin käyttää apuna lopullista avolinjan vaihevirtojen mittausjärjestelmää varten.

Avainsanat SmartMesh, avojohtolinja, Lora, C

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology Degree Programme

ABSTRACT

Author	Sami Pitkäkangas
Title	Wireless measuring of phase currents from overhead power line
Year	2017
Language	Finnish
Pages	59
Name of Supervisor	Jukka Matila

This thesis of mine was made for Vaspec Oy. The main objective of this thesis was to develop new solution for measuring phase currents on overhead power lines by using Lora-technology. The solution should contain needed technology to generate wireless communication between overhead power line and substation.

My thesis examined three possible technology to achieve wireless communication between overhead power line and substation, which by using one wireless technology a possible solution was made.

The test equipment used was called SmartMesh IP RF Certified Starter Kit made by Linear Dust Networks, where I made Windows control application by using C-programming language. The purpose of the application was to make wireless communication possible between receiver and central processing unit.

In my thesis I developed almost complete application for wireless communication by using devices which uses Lora technology. The application and other information in this thesis could be used as a base when making the complete solution for measuring phase currents on overhead power lines.

Keywords SmartMesh, Overhead power line Lora, C

SISÄLLYS

KUVIO- JA TAULUKKOLUETTELO

KÄSITELUETTELO

ISM	Industrial, Scientific and Medical, maailmanlaajuinen radiotaajuuskaista
BLE	Bluetooth Low Energy, vähävirtainen Bluetooth-tekniikka
EDR	Enchanced Data Rate, Bluetooth versioiden tiedonsiirron paranneltu versio
IoT	Internet of Things, esineiden internet
NWK	Network, Zigbee määritelmässä käytetty lyhenne verkkokerroksesta
AES	Advanced Encryption Standard, symmetrinen lohkosalausmenetelmä
FFD	Full function device, Zigbee laitetyyppi
RFD	Reduced funtion device, Zigbee laitetyyppi
MAC	Media Access Control, verkkolaitteet yksilöivä koodi ethernet-verkossa
RF	Radio frequency
PPP	Point-to-Point Protocol, tietoliikenneprotokolla
HDLC	High-Level Data Link Control, tietoliikenneprotokolla
DTE	Data Terminal Equipment, päätelaite
DCE	Data Circuit terminating Equipment, verkkopääte
FCS	Frame Check Sequence, kehystarkiste
COM	Communication port, sarjaliikenneportti

TDMA	Time Division Multiple Access, aikajakokanavointi
CTR	Counter, viestin salaukseen käytetty menetelmä
TSCH	Time Slotted Channel Hopping, Dust Networksin kehittämä kanavahyppelytekniikka
API	Application programming interface, ohjelmointirajapinta
CLI	Command-line interface, komentoliittymä
UDP	User Datagram Protocol, yhteydetön kommunikaatioprotokolla
FHSS	Frequency-hopping spread spectrum, taajuushyppely
DHSS	Direct Sequence Spread Spectrum, suorasekvensointi

1 JOHDANTO

Sähköverkot kehittyvät jatkuvasti ja rinnalle rakennetaan nykyisen teknologian tarjoamia uusia järjestelmiä verkon hallinnoimiseksi. Sähkölinjoja on rakennettu tuhansia kilometrejä ja niissä tapahtuvien ongelmien paikantaminen on tärkeä osatekijä verkon turvaamisen kannalta.

Nykyisin, kun sähkölinjalle kaatuu puu tai linjaan kohdistuu jokin sähkön kulkua vaarantava tekijä, tapahtumapaikalle lähetettävät korjaajat lähetetään paikan päälle korjaamaan linjaa, mutta ongelmaksi muodostuu vian paikantaminen, sillä vian paikantamiseen avojohtolinjasta voidaan joutua kulkemaan useita kilometrejä ennen kuin vika löydetään.

Tässä opinnäytetyössä haetaan ratkaisua avolinjan rinnalle rakennettavaan sulautettuun järjestelmään, jossa sensori mittaa virtaa linjalta ja välittää tiedon alaasemassa olevalle keskusyksikölle. Keskusyksikössä analysoidaan saatava data ja lähetetään eteenpäin sähköasemalle monitoroitavaksi.

Tässä opinnäytetyössä tarkastellaan datan lähetystä avolinjalta keskusyksikköön yhdellä langattomalla tekniikalla testattuna, jolloin sensoripuoli jää vielä tarkasteltavaksi laitteen jatkokehitystä varten.

2 OPINNÄYTETYÖN TAUSTA

Opinnäytetyö teetettiin Vaspec Oy:lle yhteistyössä TJK Tietolaite Oy:n kanssa.

2.1 Vaspec Oy

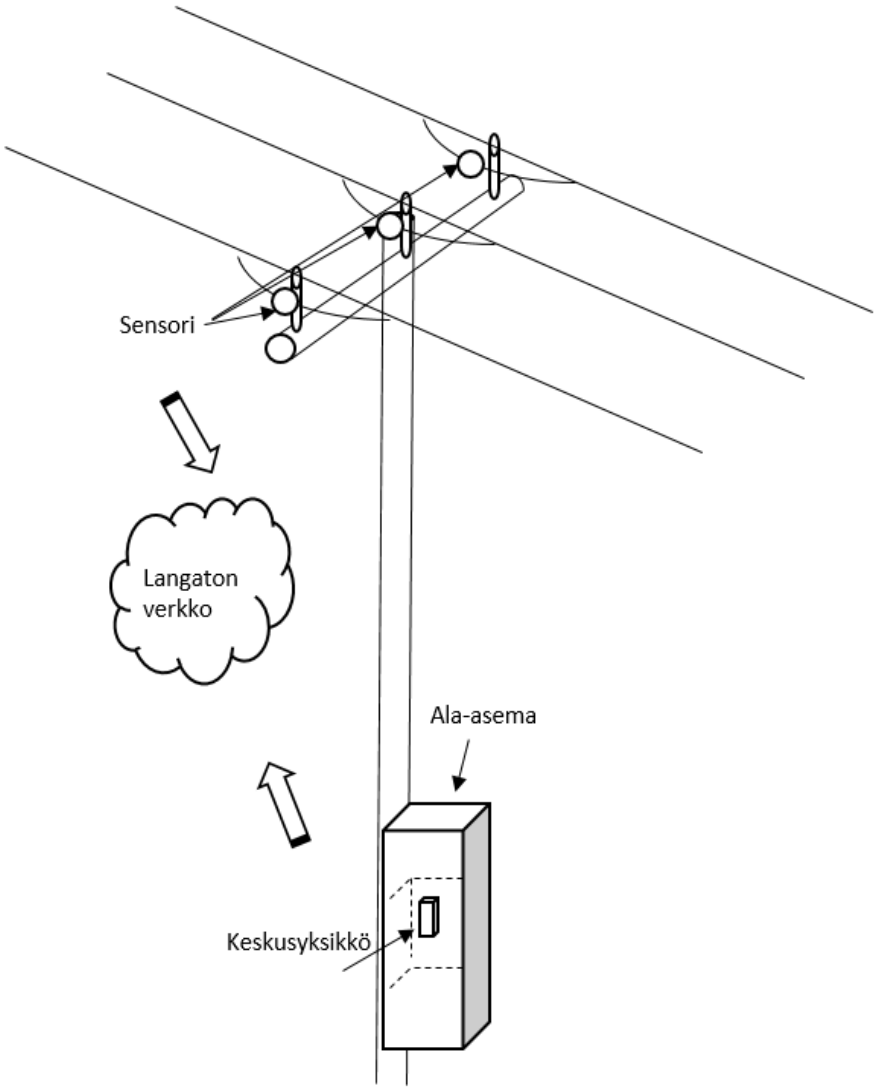
Vaspec Oy perustettiin vuonna 2011 Vaasassa ja on sähkötekniseen suunnitteluun, valmistukseen ja konsultointiin erikoistunut yritys. Yritys teettää erilaisia laitteita ja ratkaisuja sähköverkon ja siinä esiintyvien ongelmien kehitystarpeiden parantamiseksi. Yrityksen henkilömäärä tällä hetkellä on 2.

2.2 Opinnäytetyön tarkoitus

Opinnäytetyön tarkoitus on tutkia vaihevirtojen mittaamista avojohdosta ja sen välittämistä langattomalla yhteydellä ala-asemaan, josta linjalta saatu data ohjataan sähkölaitoksen keskukseen. Vaihevirtojen mittaamisella pyritään välittämään keskukselle tietoja linjalla tapahtuvista ongelmatilanteista, kuten puun kaatumisista sähkölinjan päälle. Laitteen on tarkoitus mahdollistaa linjalla tapahtuvien ongelmatilanteiden tehokas paikantaminen.

2.3 Ympäristön kuvaus

Kehitettävä laite tulee toimimaan ulkoilmassa alustavasti 3 - vaiheisessa avolinjapylväässä, jossa on yhdistettynä ala-asema. Avolinjassa, johon laite tulee, kulkee 20 kV jännite. Laite tulisi kytkeä jokaisen linjan ohituslankaan, josta se ottaa myös virtansa. Avolinjapylvään huipulla sijaitseva laite lähettää mitattuja vaihevirtoja ala-aseman keskusyksikön viereen sijoitettuun vastaanottimeen. Laitteen sijaintia kuvataan kuviossa 1.



Kuvio 1. Ympäristön kuvaus

3 LANGATTOMAT VERKOT

Tässä osiossa tarkastellaan erityyppisiä ratkaisuja ja vaihtoehtoja langattoman tiedonsiirron toteuttamiseksi avolinjalta ala-asemaan. Käsiteltyjen tiedonsiirtotekniikoiden valinta perustuu IoT-järjestelmissä käytettyihin langattomiin tiedonsiirtotekniikoihin, jotka ovat Bluetooth, LoRa ja Zigbee.

IoT-järjestelmistä puhuttaessa tarkoitetaan erilaisista laitteista koostuvia verkkoja, joissa voi olla liitettynä esimerkiksi puhelimia, tietokoneita, kodinkoneita ja muita erilaisia elektroniikkaa sisältäviä laitteita. /18/

3.1 Bluetooth

Bluetooth on avoin standardi laitteiden langattomaan tiedonsiirtoon lyhyillä etäisyyksillä. Se sai alkunsa ruotsalaisen telekommunikaatio järjestelmien valmistajan Ericssonin toimesta vuonna 1989, yrityksen alkaessa tutkia erilaisia langattoman tiedonsiirron menetelmiä. /17/

Bluetooth operoi taajuuden 2.4 GHz (ISM) lähialueella, joka on yleisesti lisensoimaton taajuusalue useimmissa valtioissa. Ainoastaan Ranskassa kansalliset rajoitukset pakottavat taajuusalueeksi 2,4465 – 2,4835 GHz. /3//17//19/

Standardi on jaoteltu virallisesti 3 luokkaan, joilla määritellään Bluetooth laitteiden suurin sallittu lähetystehon käyttö ja sen vaikutus signaalin kantavuuteen. Standardin 3 luokkaa on esitelty taulukossa 1. /1//17/

Taulukko 1. Bluetooth-luokitus /1/

Luokka	Suurin sallittu lähetysteho		Etäisyys (m)
	(mW)	(dBm)	
1	100	20	~100
2	2.5	4	~10
3	1	0	~1

Bluetoothin tiedonsiirtonopeudet ovat vaihdelleet eri versioiden julkaisujen myötä huomattavasti, opinnäytetyön puolesta kiinnostusta herätti erityisesti versiot 4.0, 4.2 ja 5.0, joita kutsutaan myös Bluetooth Low Energy (BLE) -teknologiaksi. Tekniikat on suunnattu käytettäväksi ratkaisuihin, joissa halutaan saavuttaa mahdollisimman alhainen virrankulutus. Alhaisella virrankulutuksella on kuitenkin hintansa ja tämä näkyy alhaisena tiedonsiirtonopeutena. Bluetooth-versioiden tiedonsiirtonopeudet on esitelty taulukossa 2. /17/

Taulukko 2. Bluetooth-tiedonsiirtonopeus /17/

Bluetooth-versio	**Tiedonsiirtonopeus**
1.2	0.7 Mbps
2.1	2 Mbps, EDR versiona 3Mbps
3.0	24 Mbps
4.0	0.27 Mbps
4.2	1 Mbps
5.0	2 Mbps

Bluetoothin versioista 4.0 eteenpäin käytetään nimitystä Bluetooth Smart Technology tai BLE, jonka myötä Bluetoothin käytettävyys IoT-järjestelmissä kehittyi huomattavasti. Merkittävinä uudistuksina Bluetoothissa 4.0 voidaan pitää sen vir-

rankulutuksen ja viiveen alentamista. Virrankulutuksen vähentämiseksi BLE-teknologissa on vähennetty tiedonsiirtonopeutta ja kantavuutta. /17/

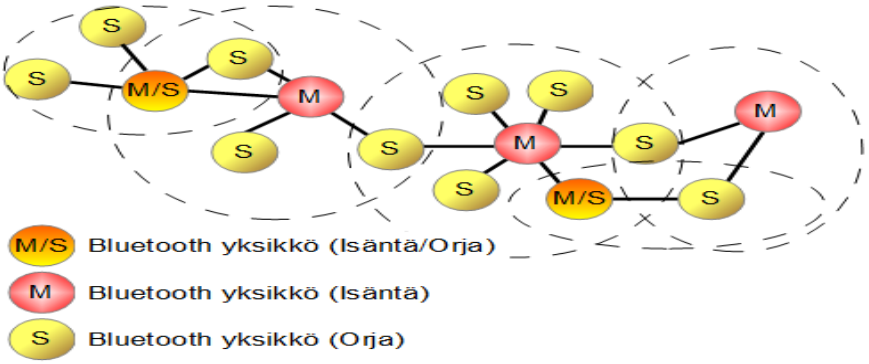
Bluetooth 4.2 ja 5.0 ovat molemmat vähävirtaisten järjestelmätekniikoiden jatkokehityksenä syntyneitä versioita. Uusimmassa versiossa Bluetooth 5.0:ssa tärkeimpinä uudistuksina on vähennetty lähetykseen ja vastaanottamiseen tarvittavaa aikaa parantamalla tiedonsiirtonopeutta, lähetettävien pakettien tiedonsiirtomäärää on lisätty ja signaalin kantosädettä on pidennetty, jopa nelinkertaiseksi. Taulukossa 3 on vertailtu klassisen Bluetooth - ja Smart Bluetooth-teknologian eroja. /17/

Taulukko 3. Bluetooth Classic vs Bluetooth Smart 4.0 vs Bluetooth Smart 5.0 /3//17//20//21/

Tekniset ominaisuudet	Klassinen Bluetooth teknologia	Bluetooth Smart teknologia 4.2	Bluetooth Smart teknologia 5.0
Etäisyys	~100 m	>100m	400m
Tiedonsiirtonopeus ilmassa	1-3 Mbit/s	1 Mbit/s	2 Mbit/s
Sovelluksen tiedonsiirtonopeus	0.7-2.1 Mbit/s	0,27 Mbit/s	-
Aktiivinen slave	7	Ei määritelty. Toteutuksesta riippuvainen	-
Turvallisuus	56/128-bit	128-bit AES, jossa CTR- mode CBC-MAC	-
Vakaus	Adaptiivinen nopea taajuushyppely, FEC, nopea ACK	24-bit CRC, 32-bit viestin eheyden tarkistus	-
Viive (yhdistämättömästä tilasta lähetykseen)	~100 ms	6 ms	-
Pienin datan	100 ms	3 ms	-

lähetysaika			
Äänen käyttö mahdollista	Kyllä	Ei	-
Verkkotopologia	Tähti	Tähti	-

Bluetooth-laitteiden muodostama verkko on tyypiltään pikoverkko. Pikoverkossa yksi muodostettu verkko voi ylläpitää kahdeksaa laitetta ja laitteiden määrää verkossa voidaan lisätä yhdistelemällä verkkoja keskenään, jolloin eri verkot voivat kommunikoida keskenään. Tätä pikoverkkojen yhdistelmää kutsutaan nimellä scatternet (**Kuvio 2.**). /17/



Kuvio 2. Usean pikoverkon yhdistelmä, scatternet

Scatternet nostaa verkossa olevien laitteiden kapasiteetin 255:een, mikä tapahtuu jättämällä laitteita sisältäviä pikoverkkoja parkkimoodiin, jolloin laite voi väliaikaisesti lähteä pikoverkosta, kun se on aktiivinen toisessa. /17/

3.2 Zigbee

Zigbee on standardi, jota valvoo Zigbee Alliance. Se pohjautuu pitkälti IEEE 802.15.4 Low-Rate Wireless Personal Area Network-standardiin ja tähtää yksinkertaisempiin ja halvempiin ratkaisuihin kuin mitä Bluetoothissa on käytetty. Operointitaajuutena käytetään ISM – taajuusalueella olevia kaistoja. /2/

Zigbee soveltuu hyvin kotiautomaatioon, mittaus- ja hallintajärjestelmiin, joissa voidaan käyttää halpoja ja pieniä mikrokontrollereita. Tiedonsiirtonopeudet ovat matalia ja virrankäyttö vähäistä. /2/

Zigbee-verkko tukee tähti-, puu- ja yleistä meshtopologiaa. Verkko koostuu yleensä kolmesta eri laitteesta; koordinaattorista, reitittimestä ja päätelaitteista. Koordinaattorin tehtävänä on uuden lähiverkon luominen, jolloin se toimii käytännössä koko verkon perustana. Koordinaattoreita ei voi olla yhdessä verkossa useampia kuin yksi. Reitittimen tehtävänä on välittää viestejä verkossa eri laitteiden välillä. Päätelaitteen tehtävänä on sisältää riittävästi toiminnollisuutta, jotta se voi kommunikoida koordinaattorin tai reitittimen kanssa. Päätelaite ei voi välittää dataa toisilta laitteilta. /2/

IEEE 802.15.4 -standardi määrittelee kaksi laitetyyppiä FFD, eli täyden toiminnollisuuden laite (Full Function Device) ja RFD, rajoitetun toiminnollisuuden laite (Reduced Function Device). FFD-laite täyttää kaikki standardin vaatimat ominaisuudet, jolloin se voi toimia koordinaattorina. Tämä vastaa käytännössä Zigbee-koordinaattoria ja toimii samalla tavalla muodostaen verkon ja vastaamalla verkossa olevista tiedoista. RFD-laitteen toiminnollisuus ja muisti on sen sijaan minimaalista ja sen tehtävänä on keskustella FFD-laitteen kanssa, mutta RFD-laitteilla ei voida keskustella toisen RFD-laitteen kanssa, esimerkiksi haluttaessa käyttää RFD-laitetta tiedon välitykseen toiselta laitteelta. Zigbeessä vähäinen virrankulutus onnistuu siksi, että RFD-laitteet voivat viettää suurimman osan ajastaan lepotilassa. /2/

Zigbee -verkossa laitteiden havaitsemisessa käytetään 64-bittistä osoitetta tai 16-bittistä verkko-osoitetta (NWK). Laitteiden oma osoite on tyypiltään unicast ja laite odottaa, että NWK-osoite on asetettu. NWK-osoitepyyntö lähetetään kaikkiin muihin verkossa oleviin laitteisiin, jotka ovat aktiivisia. Nykyisin Zigbee on siirtynyt käyttämään laitteiden tunnistuksessa IP-osoitteita. /2/

Zigbeen verkkokerros tukee tähti-, puu- ja meshtopologioita. Tähtityypin verkossa, verkkoa hallinnoi yksi laite, Zigbee-koordinaattori. Koordinaattori on vastuussa verkossa olevien laitteiden alustamisesta ja ylläpidosta. Kaikki muut laitteet kommunikoivat suoraan koordinaattorin kanssa. Mesh- ja puutyyppisessä verkossa ZigBee-koordinaattori on vastuussa verkon muodostamisesta ja muutamien verkon avainparametrien asettamisesta, mutta verkkoa voidaan laajentaa ZigBee-reitittimillä. Puuverkossa reitittimet siirtävät dataa ja hallinnoivat viestejä käyttämällä hierarkista verkkojärjestelmää. Puuverkoissa voidaan käyttää beacon -orientoitua viestintää, jota kuvataan standardissa IEEE 802.15.4-2003. Zigbeen ominaisuuksista kerätty tieto on havainnollistettu taulukossa 4. /2/

Taulukko 4. Zigbee ominaisuudet /2/ /3/

ZigBee ominaisuudet	
Taajuus	2.4 GHz, 868 MHz, 915 MHz
Virrankulutus	30 mW
Etäisyys	10–75 m
Tiedonsiirtonopeus	25-250 Kbps
Verkkotopologia	Mesh, ad hoc, tähti
Turvallisuus	128-bit salaus
Heräämisaika	15 ms

3.3 LoRa

LoRa (Long Range) on LoRa Alliancen kehittämä fyysinen kerros LoRaWAN-tiedonsiirtoprotokollalle. Tämä fyysinen kerros koostuu laitteista, signaalinkäsittelystä, modulaatiosta ja radiotiestä.

LoRan kanssa käytössä oleva tiedonsiirtoprotokolla LoRaWAN eli Low Power Wide Area Network (LPWAN) on avoin standardi, joka on tarkoitettu langattomille laitteistoille paikallisiin, maanlaajuisiin ja kansainvälisiin verkkoihin. Standardin kehitystä valvoo LoRa Alliance, joka koostuu monesta eri yrityksestä. Standardin avulla pyritään saavuttamaan kaksisuuntainen, turvallinen, liikuteltava ja paikannettava verkkoliikenne. /4/ /5/

LoRaWAN-verkkoarkkitehtuuria käytettäessä sovelletaan usein tähtitopologiaa, jossa yhdyskäytävinä toimivat ns. läpinäkyvät sillat. Nämä sillat välittävät viestejä päätelaitteiden ja keskusverkkopalvelimien välillä. Yhdyskäytävät yhdistyvät verkkopalvelimeen käyttäen perinteistä IP-yhteyttä, kun taas päätelaitteet käyttävät langatonta yhden hypyn kommunikointia yhteen tai useaan yhdyskäytävään. Kaikki päätepisteen kommunikaatio on käytännössä kaksisuuntaista, mutta tukee tarvittaessa myös eri massajakeluviestejä, kun halutaan vähentää ilman kautta kulkevaan viestintään käytettävää aikaa. /4/ /5/

Kommunikaatio LoRaWAN–verkossa on jaettu eri taajuuksiin ja eri tiedonsiirtonopeuksiin. Tiedonsiirtonopeus riippuu laitteiden etäisyyksistä sekä lähetettävästä datamäärästä. Tiedonsiirtonopeus verkossa vaihtelee 0.3 - 50 kbps välillä. Tiedonsiirronaikaisen häiriöiden estoon käytetään hajaspektritekniikkaa, jossa data pyritään levittämään laajemmalle taajuuskaistalle kuin mitä se vaatisi. Tämä estää signaalin katoamisen kokonaan, sillä esimerkiksi sähkömagneettisen häiriön sattuessa, viestistä katoaa vain kyseisellä taajuudella oleva osuus ja muilla taajuuksilla olevat osuudet säilyvät ja ovat näin ilmaistavissa vielä ymmärrettävästi. /4/ /5/ /6/

LoRaWANissa eri päätelaitteet on eritelty kolmeen eri luokkaan; A-, B- ja C -luokan laitteet. A-luokan kaksisuuntaiset päätelaitteet toimivat siten, että jokaista lähetystä seuraa kaksi lyhyttä vastaanottoikkunaa. Päätelaitteiden määräämä lähetyspaikka perustuu laitteen omiin viestintätarpeisiin. Tämä toiminto on tarkoitettu vähävirtaisiin päätelaitejärjestelmiin, jotka tarvitsevat ”downlink”-kommunikaatiota palvelimelta pian sen jälkeen, kun päätelaite on lähettänyt ”uplink”-lähetyksensä. Muina aikoina ”downlink” kommunikoinnin palvelimelta täytyy odottaa seuraavaa ”uplink” lähetystä. /4/ /5/

B-luokan kaksisuuntaiset päätelaitteet, joissa on ajastetut vastaanottopaikat avaavat ylimääräisiä vastanottoikkunoita lisäyksenä A-luokan satunnaiseen lähetysikkunaan. Laitteet saavat yhdyskäytävästä ylimääräisen aikasynkronoidun Beacon-viestin, jonka jälkeen ne avaavat omat vastaanottoikkunansa. Tällä tavalla palvelin saa tiedon, milloin päätelaite on kuuntelutilassa. /4/ /5/

C-luokan kaksisuuntaiset päätelaitteet, jossa on maksimimäärä vastaanottopaikkoja eroaa muista luokista siten, että päätelaitteissa on lähes jatkuvasti auki vastaanottoikkunoita. . LoRan ominaisuuksista kerätty tieto on havainnollistettu taulukossa 5. /4/ /5/

Taulukko 5. LoRaWAN ominaisuudet

LoRaWAN ominaisuudet	
Taajuus	868MHz, 433 MHz
Virran kulutus	Ei määritelty, jotkin ratkaisut mahdollistavat yli 10 vuoden akun käyttöiän.
Etäisyys	15 km
Tiedonsiirto nopeus	0.3 – 50 kbps
Verkkotopologia	Star – of - stars
Turvallisuus	128-bit AES-salaus

4 DUST NETWORKS™, TSMP

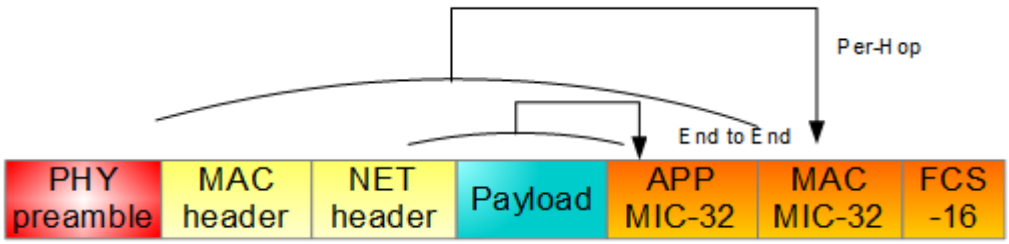
TSMP (Time Synchronized Mesh Protocol) on Dust Networks™:n kehittämä tiedonsiirtoprotokolla, itsestään organisoituville langattomien laitteiden verkoille. Näitä laitteita kutsutaan nimellä mote. Tämän protokollan mukaisten laitteiden tarkoitus on pysyä synkronisoituna keskenään ja kommunikoida tarkoin aikavälein. Tällainen kommunikaatio mahdollistaa laitteiden toimivuuden erittäin matalalla virrankulutuksella.

Protokolla on suunniteltu toimimaan erittäin häiriöisessä ympäristössä eli ympäristössä, jossa eri taajuuksilla toimia laitteita on useita. Häiriöiden välttämiseen käytetään kanavanvaihtoja, jossa TSMP-paketit lähetetään eri kanavilla, riippuen lähetyksen ajankohdasta.

TSMP mahdollistaa luotettavan, matalan virrankulutuksen ja suojatun kommunikaation langattomassa Mesh-verkossa. Alun perin protokolla suunniteltiin Wireless Hart™-standardia varten teollisen automatisoinnin tarpeisiin. TSMP mahdollistaa eri laitteiden synkronisoinnin usean hypyn verkoissa muutamassa sadassa mikrosekunnissa, mikä mahdollistaa törmäysvapaan, parillisen ja broadcast-kommunikaation aikataulutuksen vastaamaan tarvittavaan liikenteeseen laitteiden välillä, kun hypitään kanavalta toisille. Viivettä ja luotettavuutta voidaan parantaa lisäämällä virrankulutusta.

4.1 TSMP-paketin rakenne

TSMP-paketin rakenne koostuu headerista, payloadista ja trailerista. Paketit sisältävät eri kenttiä, joissa tunnistetaan lähettävä laite, määritellään vastaanottaja, varmistetaan turvattu viestin lähetys ja välitetään tietoa luotettavuudesta ja laadusta. IEEE 802.15.4-standardissa määritellään paketin suurin koko 127 B, joista TSMP käyttää 47B operoimiseen ja 80 B lähetettävään dataan. /7/



Kuvio 3. TSMP – paketin rakenne /16/

Kuvion 3 mukaisesti paketti koostuu 7 osasta, joista ”PHY”-osiossa määritellään kehyksen alku ja loppu, sekä pituus, osiota käytetään RF (radio frequency) synkronointiin. MAC-osio sisältää jokaiseen hyppyyn tarvittavat osoite- ja ajoitustiedot, laitteen tunnuksen verkossa, sekä tiedot synkronisoinnista ja liittymisestä. NET-osiosta löytyy paketin prioriteetti ja päätepisteiden osoite- ja reititystiedot. Payloadissa lähetetään salattu data, joka sisältää esimerkiksi lähetettävän käskyn tai mitatun datan. APP-osiossa tarkistetaan päätepisteiden välisen datan 32-bittinen eheys, MAC-osiossa tarkistetaan koko paketin hyppykohtainen 32-bittinen eheys ja FCS:llä suodatetaan vaurioituneet paketit IEEE 802.15.4-standardin mukaisella 16-bittisellä kehyksien tarkistussummalla. /7//16//22/

4.2 TSMP-verkon käyttäytyminen

TSMP-verkossa laitteelta laitteelle kommunikointi on jaettu aikaikkunoihin. Aikaikkunoista käytetään nimitystä aikasolu. Monen aikasolun yhdistelmällä saadaan aikaan kehys, joka toistuu verkossa koko verkon ”eliniän” ajan. Kehyksen pituus lasketaan soluina ja on näin asetettavissa oleva parametri, joka aiheuttaa eräänlaisen uudistumisajan verkossa. Lyhyellä kehyksellä voidaan lyhentää uusiutumisaikaa, joka lisää käytössä olevan kaistan hyötyä ja samalla virrankulutus kasvaa. Vastaavasti, pidentämällä kehyksen pituutta, säästetään virrankulutuksessa ja käytetään pienempää kaistanleveyttä. /7//16/

TSMP perustuu vanhempaan TDMA (Time Division Multiplexing Access)-aikajakokanavointiin ja sen yksi tärkeimmistä elementeistä on ajan synkronointi kaikkien verkossa olevien laitteiden välillä. Laitteiden täytyy jakaa sama ”ajan

taju” verkossa, jotta ne voivat keskustella, kuunnella ja olla lepotilassa oikeassa ajankohdassa. Tällä varmistetaan vähäinen virrankulutus järjestelmissä, joissa käytetään virransaamiseksi, esimerkiksi akkua tai paristoja. Laitteiden samalla ajan tajulla mahdollistetaan monia eri hyötyjä verkossa, kaistoja voidaan varata, jolla saavutetaan luotettavat lähetykset ja poistetaan kokonaan laitteen mahdollinen omien lähetyksien häiriköinti. Lähettävät laitteet voivat vaihdella taajuuksia jokaisen lähetyksen kohdalla ja vastaanottava laite voi pysyä omalla taajuudellaan. /7//16//22/

TSMP-laitteet verkossa toimivat verkossa kolmella tavalla; lähettämällä viestejä toiselle laitteelle, vastaanottamalla tietoa toiselta laitteelta ja toimimalla rajapintana anturille tai prosessorille. Kaikkina muina aikoina laitteet ovat lepotilassa, mikä ehkäisee langattomassa viestinnässä suurimman osan virrankulutuksesta, joka aiheutuu radiomaston päälläolosta. /16/

Verkossa TSMP jakaa langattoman yhteyden aikaan ja taajuuteen. Tämä mahdollistaa vahvan virheensietokyvyn tavallisia RF-liikenteeseen kuuluvia häiriöitä vastaan ja samalla kaistankäytöstä tulee tehokkaampaa. RF-liikenteeseen liittyviä haasteita on pyritty ohittamaan käyttämällä FHSS(Frequency Hopping Spread Spectrum)- ja DSSS(Direct Sequence Spread Spectrum)-tekniikkaa. FHSS-tekniikassa pyritään hyppimään kanavilta toisille, jolla pyritään vähentämään häiriöiden määrää. DSSS-tekniikalla jaetaan sanoma pieniin osiin ja lähetetään koko kaistanleveydellä yhtenä signaalina. Yhdistelemällä näitä tekniikoita, TSMP:ssä on saatu aikaiseksi hyvä häiriöiden sietokyky verkossa. /7//16/

Verkon yhtenä tärkeimpänä ominaisuutena voidaan pitää sen itsestään organisoituvuutta. Verkossa olevilla laitteilla on riittävän suuri älykkyys tunnistamaan verkossa olevat muut laitteet, mittamaan RF-signaalin vahvuus, synkronointi ja taajuushyppelyn tiedon kerääminen ja yhteyksien muodostaminen muiden verkossa olevien laitteiden kanssa. Nämä ominaisuudet ovat suurin syy siihen, miksi TSMP-verkossa käytetään mesh-topologiaa. /16/

Verkossa käytetään täysin redundanttia mesh-reititystä, jolla tarkoitetaan lähetyksien kahteen kertaan lähettämistä eli lähetettävästä paketista lähetetään ns. toisteinen. Tällä pyritään parantamaan signaalin läpimenoa RF-ympäristöissä. Redundantti reititys tarvitsee toimiakseen kahta eri tilaa, jotka ovat signaalin lähetys eri reittiä pitkin ja signaalin lähetys myöhemmin uudelleen. TSMP:ssä tämä on mahdollistettu sallimalla verkon laitteiden etsiä useampia isäntälaitteita ja sallimalla niiden muodostaa yhteys useampaan isäntälaitteeseen. /7//16/

Verkon turvallisessa viestinnässä käytetään viestinsalausta, varmennetta ja eheyden tarkistusta. Salauksessa käytetään teollisen tason 128-bittistä symmetristä avainta salaamaan päätepisteiden välillä lähetettävä data. Laitteet, jotka jakavat saman avaimen, kommunikoivat käyttäen CTR(Counter)-tyyppistä salakirjoitusta. CTR- tyyppisessä salakirjoituksessa lähettäjä ja vastaanottaja ovat yhteydessä jaettuun laskimeen, jossa lasketaan uusi yhteinen luku, jota käytetään viestin salaamisessa. /7//16//22/

Koska salauksella varmistetaan viestin luotettavuus, varmennetta käytetään lähettävän laitteen tunnistamiseen. Jokaisen paketin lähettäjän verkossa lähettämisen varmentamiseksi TSMP:ssä käytetään paketin lähettäjän osoitetta suojattuna 32-bittisellä MIC(message integrity code)-koodilla. Jokaisessa paketissa on kaksi MIC-koodia, jolla varmennetaan lähettäjä. Nämä koodit sisältävät päätelaitteiden välisen lähettäjän osoitteen varmentamisen verkkokerroksen MIC-koodilla ja laitteelta laitteelle lähettäjän osoitteen varmistettuna MAC-kerroksen MIC-koodilla. Samalla MIC-koodeilla, joilla varmistetaan lähettäjän osoite, varmistetaan lähetettävän paketin eheys. /7//16//22/

5 DUST NETWORKS™ LINEAR TECHNOLOGY

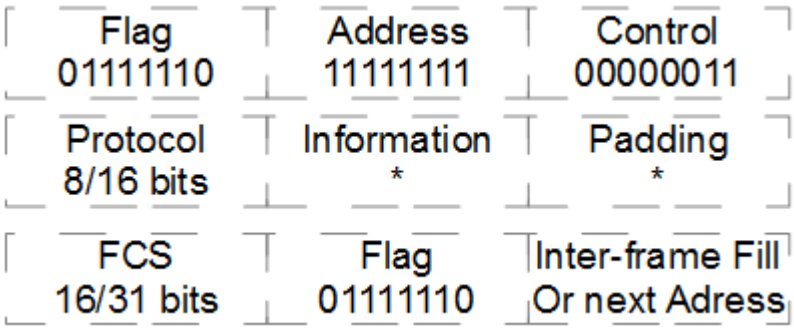
RFC 1662 vaikutus Dust NetWorks™ Linear Technology tuotteisiin, näkyy niiden sarjaliikennetyyppisessä ohjauksessa, jossa viestit välitetään RFC 1662-standardia mukailevina HDLC (High-Level Data Link Control)-kehyksinä. Standardista on otettu myös käyttöön siihen liittyvä FCS (Frame Check Sequence) laskenta. Tämän osion tarkoitus on tarkastella RFC 1662-standardia, sekä Dust NetWorks™ Linear Technologyn verkko- ja tiedonsiirtotekniikkaa.

5.1 RFC 1662-standardi

Standardi kuvailee PPP(Point-to-point)-protokollaa HDLC-tyyppisessä kehyksessä. Tätä standardia käytetään mm. Dust Networks™ Linear Technologyn tuotteissa API(Application interface)-pohjaiseen kommunikointiin laitteiden kanssa. /8/

Fyysisenä kerroksena tiedonsiirtoon käytetään päätelaite (DTE) ja verkkopääte-(DCE)rajapintoja, esimerkiksi RS -232-E, RS – 422, CCIT V35. Käytännössä PPP-protokollalla halutaan tarjota kahden laitteen väliselle tietoliikenteelle kaksisuuntainen kommunikaatio. /8//22/

Tiedonsiirrossa lähetettävien viestien kehyksien formaatti on 8- bittisistä kentistä koostuva kokonaisuus (**Kuvio 4.**).



Kuvio 4. RFC 1662 HDLC-kehys /8/

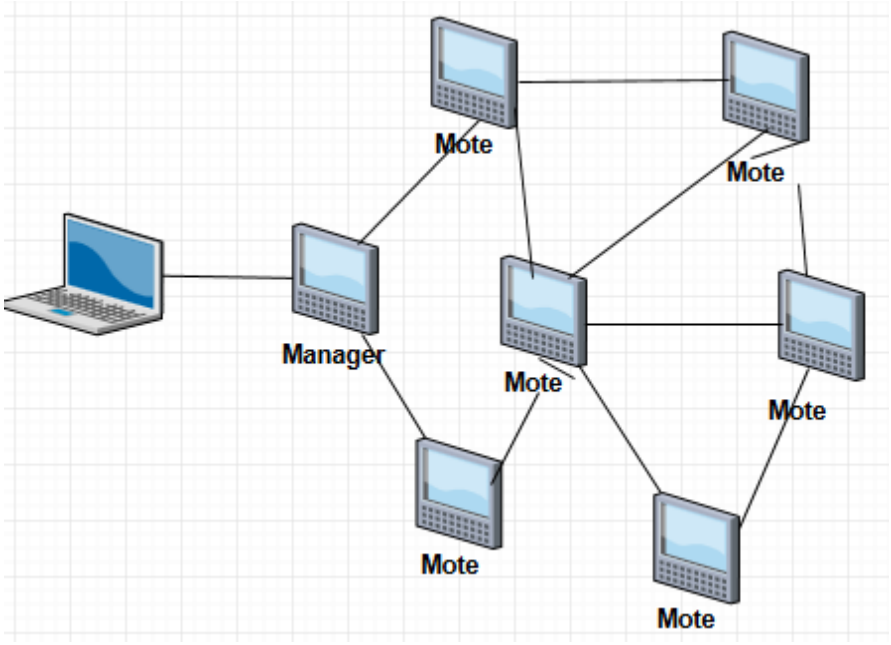
Kuviossa 4 kuvataan HDLC-tyyppisen kehyksen rakennetta. Kehyksen lippu-(flag)sekvenssi alkaa ja loppuu aina samalla tavalla heksadesimaalilukuun 0x7E, joka on kuvassa binäärisenä. Eri toteutuksissa tarkistetaan jatkuvasti tätä lippua, koska sitä käytetään kehyksen synkronointiin. Vain yksi lippusekvenssi lähetetään kahden kehyksen aikana. Lähetettäessä kaksi lippusekvenssiä kahden kehyksen välissä, syntyy tyhjä kehys, joka poistetaan sekvenssistä, eikä sitä lasketa FCS-virheeksi. Osoitteen sisältävässä kehyksessä lähetetään 8-bittinen sekvenssi kaikkien laitteiden osoitteisiin. Kaikkien eri asemien osoitteet pitää aina tunnistaa ja vastaanottaa. Muiden yksittäisten laitteiden osoitteet voidaan määritellä myöhemmin tai tarpeen vaatiessa. Kehykset, jotka sisältävät tunnistamattomia osoitteita, poistetaan. Ohjauskehyksessä lähetetään 8-bittinen sekvenssi heksadesimaalina 0x03, jolla kerrotaan, että laitteita halutaan ohjata. Kehykset, jotka sisältävät väärän sekvenssin, poistetaan. FCS-kehys voidaan lähettää kahtena 8-bitin sekvenssinä tai neljänä 8-bittisenä sekvenssinä. FCS-kehyksessä lasketaan kaikki osoite -, ohjaus -, protokolla -, tieto - ja täytekehyksien sisältämät bitit. /8/

5.2 SMARTMESH IP

SmartMesh IP-verkko on Dust Networkin kehittämä älykäs langaton verkko, joka on suunnattu sulautettujen järjestelmien rinnalle.

Verkko koostuu itsestään organisoituvan Mesh-verkon muodostavista laitteista, joita kutsutaan Dust Networksin dokumentaatiossa moteiksi. Motet keräävät ja

välittävät dataa verkon managerille, jolla voidaan valvoa ja hallinnoida verkon suorituskykyä, suojausta ja tiedon välitystä pääohjelman kanssa (**Kuvio 5.**). /9/



Kuvio 5. SmartMesh IP kuvattuna /9/

SmartMesh IP-verkossa kommunikoidaan TSCH-linkkikerrosta käyttäen. TSCH on kehitetty Linear Dust Networkin toimesta. Siinä motet pystyvät synkronoimaan millisekunneissa. Verkon aika on järjestelty aikasoluiksi, jolla mahdollistetaan yhteentörmäyksistä vapaa paketin lähetys ja lähetyskohtainen kanavahyppely. /9/

SmartMesh IP-järjestelmän vahvuuksia on alhainen virrankulutus, hyvä verkon vakaus, Ipv6-osoitteiden käyttämisen mahdollisuus, matalan viiveen tila ja kattava suojaus asetuksien hallinnointiin. Verkossa olevat laitteet voivat toimia paristoilla, virrankeräyksellä tai kaapelilla. Verkon luotettavuudeksi ilmoitetaan > 99.9 %, jopa kovissa RF-ympäristöissä. IPv6-osoitteissa yhdistyy 6LoWPAN ja IEEE 802.15.4e /9/

5.3 HDLC kehys SMARTMESH IP

Dust Networks Linear Technology tuotteet käyttävät API-pohjaiseen kommunikointiin RFC 1662-standardin mukaista HDLC-kehystä, pienellä eroavaisuudella RFC-standardissa kuvailtuun versioon.

Tässä HDLC-kehyksen variaatiossa alku-ja loppulippu ovat samoja kuin RFC 1662:ssa eli heksadesimaalina 0x7E ja paketit sisältävät 16-bittisen CRC-CCITT FCS-sekvenssin. Erona RFC 1662 mukaiseen kehykseen on HDLC-ohjaus- ja osoitekenttien puuttuminen. Kokonaisuudessaan HDLC-kehys kuvattuna kuviossa 6. /15/

Start Flag (Byte 0)	HDLC Payload (Bytes 1-n)	FCS (Bytes n+1,n+2)	End Flag (Byte n+3)
0x7E	HDLC escaped API payload	(2 bytes)	0x7E

Kuvio 6. HDLC Frame SmartMesh IP /15/

HDLC Payload kehyksen sisältö koostuu kahdesta osiosta, API header ja API Payload, jotka sisältävät kommunikointiin tarvittavat kentät. Kentistä Api header sisältää Linear Technologyn dokumentaation mukaiset käskyille määritellyt arvot, viestin pituuden sekä lipun. API Payload-kenttä sisältää viestien kuittauksen tai itse lähetettävän viestin. Kentät havainnollistettuna kuviossa 7.

API Header	API Payload
Command | Length | Flags	Response code| Message Payload

Kuvio 7. HDLC Payload SmartMesh IP /15/

5.4 HDLC-paketin muodostaminen SmartMesh IP

Tässä osiossa suoritetaan HDLC-viestin muodostaminen SmartMesh IP motelle käyttäen apuna käskyä sendTo, jolla lähetetään viesti toiselle laitteelle.

Aluksi täytyy selvittää Linear Dustin dokumentaatiossa määritelty käskyjen sisältö sendTo-toiminnolle, joiden avulla kootaan kuviossa 7 oleva API payload-kenttä. Dokumentaatiossa määritellyt parametrit on kuvattuna taulukossa 6. /15/

Taulukko 6. SendTo parametrit

Parametri	**Tyyppi**	**Kuvaus**
socketId	INT8U	Socket ID
destIP	IPV6-osoite	Vastaanottajan IPV6-osoite. Sovelluksille, jotka eivät lähetä verkon isännälle tulee käyttää managerille määritettyä osoitetta (FF02::2)
destPort	INT16U	Vastaanottajan porttinumero
serviceType	INT8U	Palvelun tyyppi
priority	INT8U	Paketin prioriteetti. Matalampi paketin prioriteetti, odottaa korkeampi prioriteettisten pakettien lähetystä
packetId	INT16U	Käyttäjän asettama paketin numero txDone-ilmoituksen saamiseksi
payload	INT8U[]	Paketin sisältämä data

Taulukossa 6 Socket id-kentän sisältö saadaan suorittamalla motelle käskyt open socket ja bind socket. Open socket-toiminnolla avataan motelle päätepiste verkossa kommunikointia varten. Motelle muodostuvan socketin tiedonsiirtoprotokollaksi voi valita vain UDP(User Datagram Protocol)-protokollan. Käskyjen suorittamisen jälkeen mote luo socketin ja lähettää vastaukseksi socket id:n, joka on asetettu alkamaan numerosta 22. /15/

Kun halutaan avata useampia päätepisteitä, niin socket id:n numero jatkuu numerosta 22 eteenpäin. Tälle muodostuneelle socket id:lle suoritetaan käsky bind

socket, joka sitoo moten haluttuun päätepisteeseen ja käskyä suoritettaessa päätepisteelle asetetaan portti, jonka kautta kommunikaatio ohjataan. Porttia valittaessa, on hyvä valita portin numero väliltä 61624 – 61631, sillä nämä porttinumerot mahdollistavat suurimman kapasiteetin viestien lähettämisessä. /15/

Vastaanottavan laitteen IPV6-osoitteena käytetään dokumentaatiossa ennalta määriteltyä IP-osoitetta. Palvelun tyypin (service type) vaihtoehdoiksi on ainoastaan käytettävissä taajuustyyppinen palvelu. Lopuksi käyttäjä valitsee paketille prioriteetin, jolla määritellään suoritusajankohta paketissa, valitsee paketille annettavan tunnistenumeron (packet id) ja valitsee lähetettävät arvot (payload). Taulukossa 7 on havainnollistettu koottu API Payload kenttä arvoineen. /15/

Taulukko 7. SendTo parametrit API Payload /15/

Parametri	Numeroarvo	HDLC UART parametri	Kuvaus
Socket Id	22	0x16	Open socket-käskyn suorittamisesta saatu parametri
destIP	ff02:02	0xff 0x02 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x02	Managerin oletusarvoinen osoite
destPort	61624	0xF0 0xB8	Bind socket-käskyllä asetettu portin numero
serviceType	0	0x00	Ei muita vaihtoehtoja
priority	2	0x02	Valittu arvo

packetId	1234	0x04 0xD2	
payload	00	0x00	

Kun tarvittavat API payload-kentän parametrit on saatu muodostettua, aloitetaan laskemalla 8-bittisten parametrien määrä, jolla selvitetään kuviossa 7 olevan API header-kentän tarvitsema lähetettävän viestin pituus, etsitään komennolla dokumentaatiossa määritelty komennon arvo ja valitaan API headeriin haluttava lippu. Lipun avulla voidaan valita onko käsky ilmoitusluontoinen vai halutaanko laitteella suorittaa jokin käsky. Lopuksi lasketaan FCS-tarkistussumma ja lisätään kuvoissa 6 jo havainnollistetut alku- ja loppuliput. Tämän jälkeen voidaan muodostaa lopullinen lähetettävä viesti, joka on kuvattuna taulukossa 8. /15/

Taulukko 8. SendTo koko HDLC viesti koottuna

HDLC kenttä	**Sisältö**	**Parametrit**	**Kuvaus**
Start Flag		0x7E	**Kuvio 6**
API header	Command, length, Flag	0x18 (Send To cmdId) 0x18 (Payload kentän pituus) 0x00 (Request Id)	Sisältää dokumentaatiossa määritellyt arvot käskyille ja toiminnoille

API payload	Socket Id, destIP, destPort, serviceType, priority, packetId, payload	0x16 (Määritetty socket Id) 0xff 0x02 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x02 (IP-osoite 16-bittiä) 0xF0 0xB8 (asetettu portti) 0x00 (Bandwidth) 0x02 (Priority) 0x04 0xD2 (Packet Id) 0x00 (Payload)	Määritelty taulukossa 7
FCS	Laskettu 16 –bit CRC –CCITT API-headeristä ja API-payloadista	0x4F 0x17	
End Flag		0x7E	**Kuvio 6**

5.5 HDLC saapuvan viestin tulkinta SmartMesh IP

Dekoodausta tarvitaan kun HDLC-viesti pitää tulkita. Dekoodaus suoritetaan viesteille, jotka alkavat ja loppuvat lippuun 7E. SmartMesh IP -verkossa laitteen suorittaessa sille annetun käskyn, se saa vastaukseksi ilmoituksen suoritetusta toiminnosta ja sen tilasta, saatiinko käsky suoritettua vai oliko suoritetun käskyn rakenteessa jokin virhe. Onnistuneesta saapuneesta viestistä esimerkki taulukossa 9. /15/

Taulukko 9. HDLC Response-viesti

Saapuva Paketti 0x7E 0x25 0x03 0x02 0x04 0xd2 0x00 0x38 0x4d 0x7E							
Start Flag 0x7E	**Notification cmd** 0x25	**Length Payload** 0x03	**Packet Flag** 0x02	**Packet Id** 0x04 0xd2	**Transmit Status** 0x00	**FCS** 0x38 0x4d	**End Flag** 0x7E

Taulukon 9 mukaisen SendTo-toiminnon (**Taulukko 8.**) jälkeen palautuvan viestin sisältö tulkitaan käyttäen Linear Dustin dokumentaatiota. Viestissä alku- ja loppuliput ovat näkyvillä selvästi 0x7E. Notifikaatio-kentässä kerrotaan onnistuiko lähetys, 0x25 tarkoittaa txDone eli lähetys tehty. Paketin pituus 0x03 kertoo parametrien määrän. Paketin lippu-kentässä 0x02 indikoi vastaus viestiä. Packet Id 0x04 0xd2 1234. Lähetyksen tila 0x00, joka vastaa lähetyksen onnistumista ja lopuksi FCS-tarkistesumma, jolla varmistetaan paketin eheys. /15/

6 TESTAUS

Tässä opinnäytetyön osiossa testattiin Fintronicilta koekäyttöön saatua laitteistoa (DC9021 SmartMesh IP RF Certified Starter Kit). Tarkoituksena oli tutkia testilaitteiston tarjoamaa toiminnollisuutta ja helppokäyttöisyyttä, sekä testata laitteiston sopivuutta käytettäväksi langattomana tiedonsiirtotekniikkana avojohtolinjassa.

6.1 Testilaitteisto

Testilaitteistoksi valittiin LoRa-tekniikka käyttävä ja TSMP-tuen sisältävä Linear Dust Networks:n valmistama testilaitteisto. Valintakriteereinä olivat aikasynkronointi, virrankulutus ja LoRa-tekniikan käyttö. Tarkoituksena oli saada testattua 3 sensorin ja 1 keskusyksikön välisen kommunikoinnin toteuttamista.

Testilaitteisto sisältää seuraavia komponentteja:

1 x DC2274A-A SmartMesh IP manager (USB) ”DC9021A”

5 x DC9018A-B SmartMesh IP RF Certified mote “DC9021A”

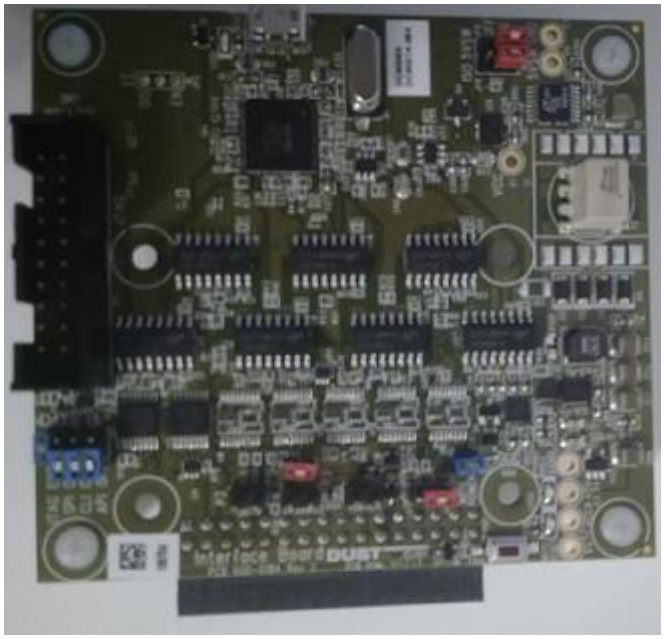
1 x DC9004A Programming Adapter

1 x DC9021A Dust Networks Eval/Dev Interface Board “DC9006A”

Testeissä käytettiin paketin sisällöstä moteja, manageria sekä Interface Boardia.

Kuvio 8. SmartMesh IP manager (USB)

Kuvio 9. SmartMesh IP RF Certified mote



Kuvio 10. Dust Networks Eval/Dev Interface Board

6.2 Laitteiston testaamisessa käytetyt ohjelmistot ja käyttöönotto

Testikäytössä käytettiin omaa kannettavaa tietokonettani, jonka käyttöjärjestelmänä toimii Windows 7 Home Premium. Testaamisessa käytetyt ohjelmistot ovat kaikki Linear Dust Networks:n tarjoamia tuki-/kehitysohjelmistoja testilaitteiston käyttäjille.

6.2.1 FTDI Serial Drivers

FTDI Serial Drivers –ajureita vaaditaan USB-porttien muuntamisessa sarjaliikennekommunikaatioon sopiviksi. Ajurit täytyy asentaa jos testilaitteiston manageria liitettäessä ei saada tietokoneen laitehallinnassa näkyviin virtuaalisia sarjaliikenneportteja (**Kuvio 8.**). Eri käyttöjärjestelmissä valmiiksi asennettujen ajurien kattavuus vaihtelee ja joissakin tapauksissa ne voivat olla valmiiksi asennettuna.

Ajurien asennus tapahtuu ensimmäisenä otettaessa laitteistoa käyttöön. Manageria liitettäessä, tietokoneen laitehallintaan ilmestyy tunnistamaton laite ja laitehallin-

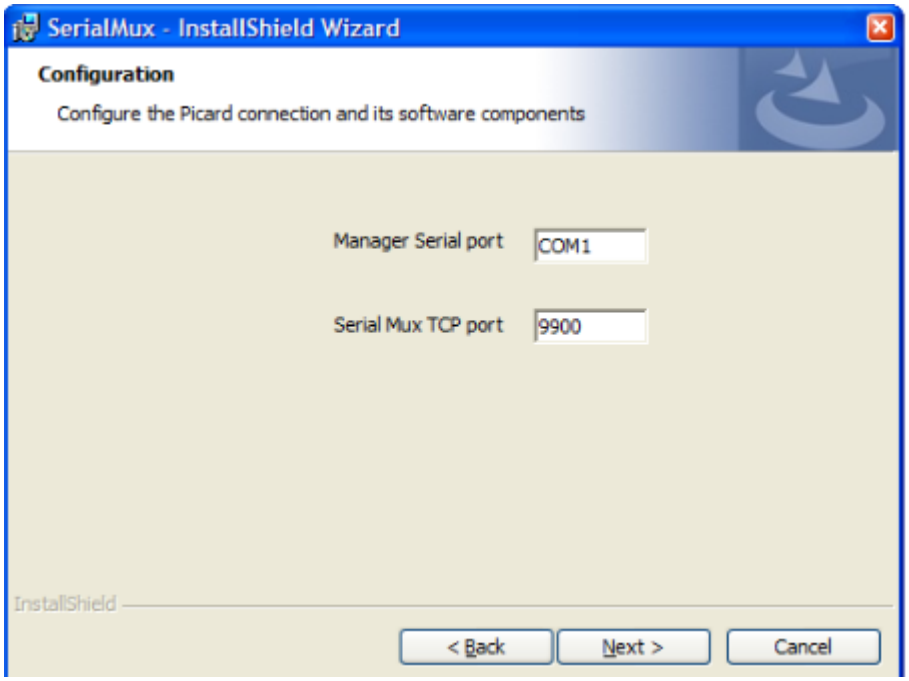
nassa tunnistamatonta laitetta oikealla hiiren näppäimellä klikattuna, saadaan avattua ohjainohjelmiston päivitysvalikko. Päivitysvalikon kautta voidaan etsiä verkosta automaattisesti päivitystä tai tietokoneesta voidaan valita manuaalisesti asennettava ajuri. FTDI-asennuspaketin ladatessa ajureita, löytyy useamman tyyppisiä, i386 ja amd64. Näistä kahdesta ajurityypistä i386-ajuri on tarkoitettu Intel-tai AMD-prosessorilla toimivaan 32-bittiseen käyttöjärjestelmään ja amd64-ajuri on tarkoitettu vastaavaan 64-bittiseen järjestelmään. Onnistunut ajurien asennus näkyy tietokoneen laitehallinnassa siten, että tunnistamaton laite häviää ja managerin liittyessä tietokoneen USB-porttiin ilmestyy laitehallintaan 4 virtuaalista sarjaporttia (**Kuvio 11**).

Kuvio 11. Windows Device manager, virtuaaliset COM-portit

6.2.2 Serial Mux

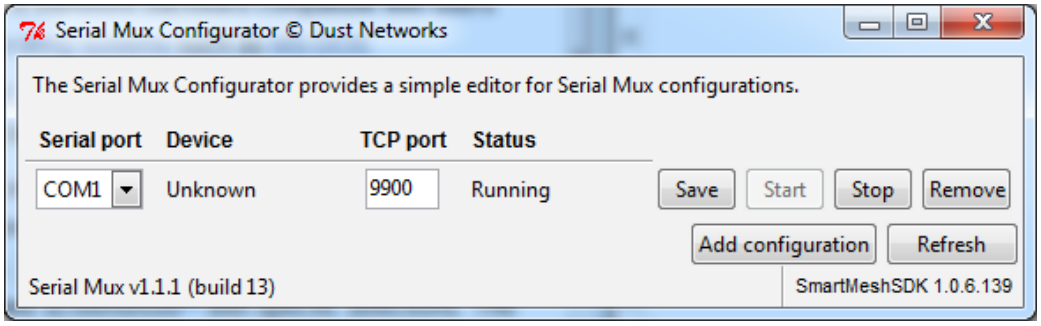
Serial Mux -ohjelmistoa käytetään mahdollistamaan monen asiakkaan kommunikoiminen manager-laitteen kanssa. Nimensä mukaisesti Serial Mux toimii multiplekserin tavoin eli voidaan ohjata useita eri signaaleita ja prosesseja kommunikoimaan yhden linjan kanssa. Testilaitteiston ohjelmista esimerkiksi Stargazer –ohjelmisto kommunikoi manager-laitteen kanssa käyttäen Serial Mux:a.

Serial Mux-lataustiedoston mukana tulee asennettaessa asennustiedosto, jonka avulla asennus on hyvin yksinkertaista. Serial Muxia asentaessa määritetään se virtuaaliportti, johon manager-laite on liitettynä ja kommunikaatioportti, joka vakiona on 9900. Portin voi määritellä itsekin, mutta se pitää valita siten, että se ei sekoitu muiden toimintojen käyttämien porttien kanssa.



Kuvio 12. Serial Mux-ohjelmisto, manager-portin määritys

Portin numero määritellään Linearin dokumentaation mukaisesti siten, että se on managerin muodostamasta 4 virtuaalisesta portista suurimman numeron saanut portti. Windows 7-käyttöjärjestelmällä tehtyjen testiasennuksien perusteella portin numero vaihtelee, jolloin kokeilemalla yhteydenottoa, esimerkiksi terminaaliohjelmaa apuna käyttäen, saadaan selville oikea portti. Mikäli määritettäessä porttia kirjoitetaan vahingossa ohjelmistoon väärä portin numero, niin ohjelmistossa on mahdollista asettaa portti uudelleen käyttäen SmartMesh SDK-tiedostopaketin mukana tulevaa ohjelmaa Serial Mux Configurator (**Kuvio 13.**). Vaihtoehtoisesti uudelleenmääritys onnistuu myös manuaalisesti kirjoittamalla suoraan Serial Muxin konfiguraatiotiedostoon ”serial_mux.cfg” haluttu portin numero ja uudelleenkäynnistämällä tietokoneen palveluista SerialMux_Default -palvelu. /11/



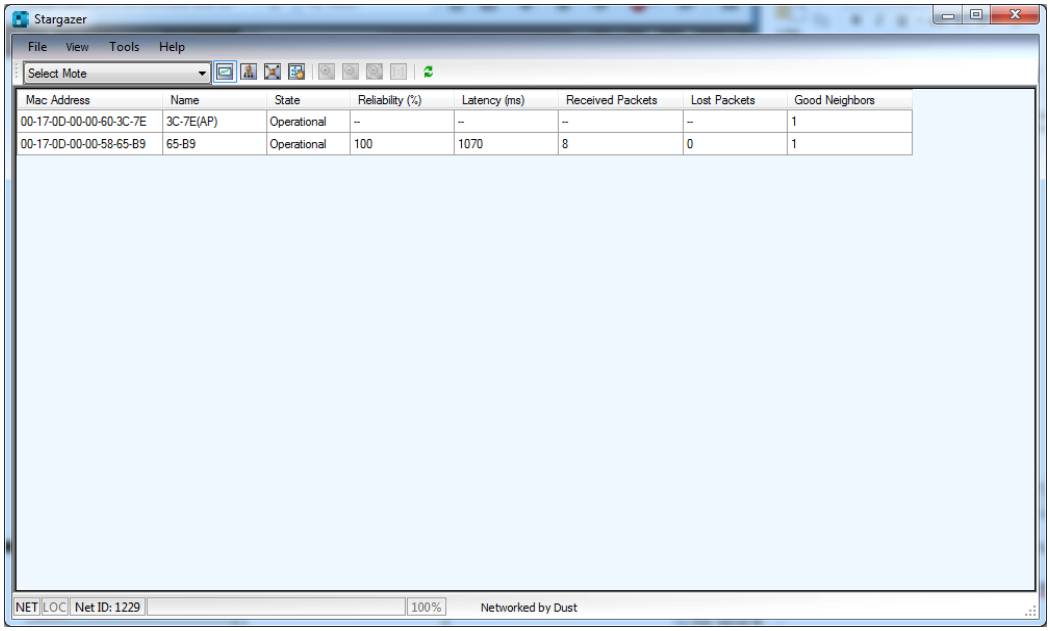
Kuvio 13. Serial Mux-konfiguraatioikkuna

6.2.3 Stargazer

Stargazer-ohjelmisto tarjoaa graafisen visualisoinnin laitteiden muodostamasta verkosta. Lisäksi sillä voi kommunikoida rajoitetuin toiminnoin verkon kanssa. Ohjelmisto tarvitsee toimiakseen FTDI-ajurien ja Serial Muxin asennuksen. Itse ohjelmiston asennuspaketissa mukana tulee asennusohjelma, jonka avulla ohjelman asennus tapahtuu asennusohjelman ohjeita noudattaen.

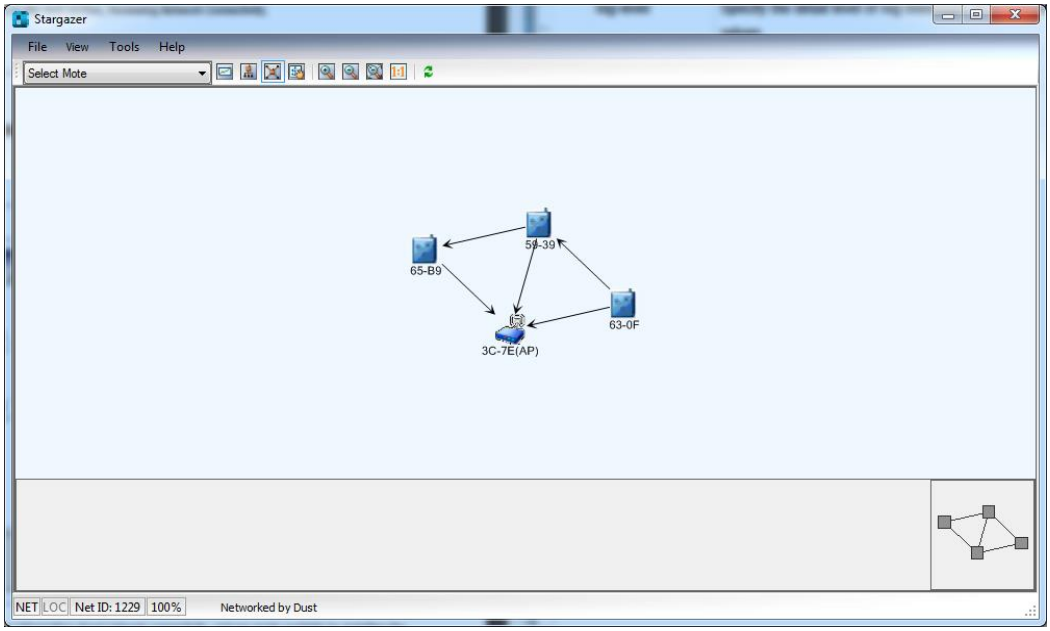
Stargazer-ohjelmistolla voidaan visualisoida laitteiden muodostamaan verkkoa 4 tavalla, joita ovat mote , hierarkinen , radioyhteyksien ja manuaalisesti asetetun tilan visualisointi.

Testauksissa käytettiin apuna kahta eri visualisointiominaisuutta, jotka olivat mote ja radioyhteyksien visualisointi. Mote-visualisoinnilla tarkoitetaan taulukkomuotoista visualisointia verkossa olevista laitteista, jossa ilmaistaan laitteen nimi, tila, yhteyden vakaus, viive, lähetetyt ja kadonneet viestipaketit ja laitteeseen liitoksissa olevat ”naapuri” laitteet. Mote-visualisointi havainnollistettu kuviossa 14. /11/



Kuvio 14. Stargazer manager 3C-7E yhdistettynä 1 x mote 65-B9

Toinen Stargazer-ohjelmiston avulla tehty visualisointi verkosta oli radioyhteyksien visualisointi. Testauksissa visualisoitiin todellista tilannetta, jossa kolme sensorilaitetta on liitettynä yhteen vastaanotinlaitteeseen. Tällä pyrittiin todistamaan avojohtopylvään kolmen sähkölinjan kommunikoinnin mahdollisuus ala-asemassa olevaan vastaanottimeen. Radioyhteys havainnollistettu kuviossa 15.



Kuvio 15. Stargazer-kuva verkosta, jossa manager + 3 motea

6.2.4 PuTTY

PuTTY on avoimen lähdekoodin telnet- ja ssh-asiakasohjelma ja pääte-emulaattori, jonka alkuperäinen kehittäjä on Simon Tatham. PuTTYä tai vastaavia ohjelmia käytettäessä on hyvä tietää, että tämän tyyppisten ohjelmien käyttö on laitonta maissa, joissa viestinnän salaus on kiellettyä. Lisätietoja maakohtaisista asetuksista löytyy listattuna esimerkiksi sivustolta cryptolaw.org. /23/

PuTTY on saatavilla Windows – ja Unix käyttöjärjestelmille ja sen asennus tapahtuu yksinkertaisesti lataamalla asennustiedosto ja seuraamalla asennusohjelman ohjeita. PuTTYä asennettaessa muita vaadittavia esiasennettuja ohjelmistotarpeita ei ole. /24/

Tässä opinnäytetyössä PuTTYä käytetään pääte – emulaattorina sarjaportin kautta kommunikoimiseen testilaitteiston kanssa. PuTTY-ohjelman valinta käytettäväksi pääte-emulaattoriohjelmana perustui työntekijän aikaisempaan käyttökokemukseen tämän tyyppisistä ohjelmista. /24/

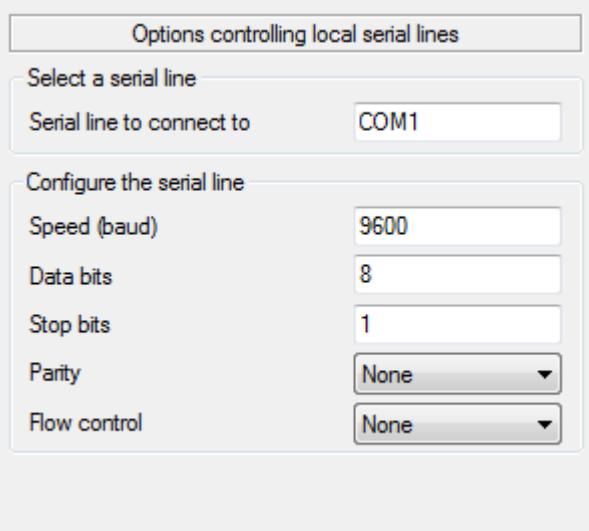
6.3 Laitteiston sopivuuden ja CLI-puolen testaus

Ominaisuuksia, joita testauksissa laitteistolta haettiin ovat; kommunikaation toimivuus, laitteiston luotettavuus ja sopivuus 3 sähkölinjan avojohtolinjaan (**Kuvio 15.**). Linear Dustin-testilaitteistossa mukana oli valmiiksi toteutettu ratkaisu TSMP:lle, joka poisti tarpeen toteuttaa se itse.

6.3.1 Manager DC2274A-A

Manageri yhdistettiin PC:hen USB:llä, jonka jälkeen manager muodosti virtuaaliset sarjaliikenneportit. Laitteeseen syttyi tietokoneeseen kiinnitettäessä sininen LED-valo, joka indikoi managerin päälläoloa ja sen yrityksiä muodostaa verkkoa. Verkkoa muodostettaessa laite lähettää jatkuvasti broadcast-viestejä yrittäen löytää verkkoon liittyviä laitteita.

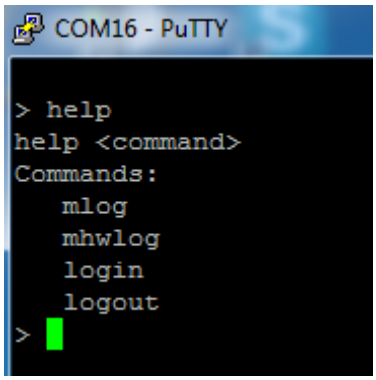
Ennen komentojen suorittamista, PuTTY vaatii konfiguroinnin sarjaliikenteelle managerin kanssa kommunikoimiseksi.



Kuvio 16. PuTTY-konfigurointi sarjaliikenteelle

Managerin muodostamaa 4:ää COM-porttia käytetään kommunikoitaessa managerin kanssa. Managerin luomat portit testeissä olivat COM8,COM9,COM10 ja COM11. Dokumentoinnin perusteella CLI-kommunikointiin tarkoitettu portinnu-

mero on toiseksi suurin eli COM10, mutta testeissä ilmeni, että portin numero saattaa vaihdella käytettäessä Windows 7-käyttöjärjestelmää. Tällöin CLI-portin numeroa voi joutua arvailla, esimerkiksi ottamalla yhteys eri COM-portteihin, jonka jälkeen yritetään vaihtaa riviä ja voidaan esimerkiksi yrittää kirjoittaa komento help, joka tuo näkyviin käytettävissä olevat käskyt (**Kuvio 17.**).



Kuvio 17. CLI-portin havaitseminen käskyllä ”help”

Manager-laitteeseen kirjauduttaessa käskyksi annetaan login user, jonka jälkeen managerin komentoja voidaan suorittaa. Käytettävissä olevat käskyt löytyvät listattuna Linearin dokumentaatiosta ja saadaan myös näkyviin käyttämällä käskyä ”help”. /12/

Kaikista managerilla olevista käskyistä yhtenä testienkannalta tärkeimmistä pidettiin sm-käskyä, jolla voidaan seurata managerin muodostaman Mesh-verkon laitteita. Tämän avulla voitiin tarkastella ja seurata verkossa olevien laitteiden tiloja, verkossa olemisen aikaa, laitteiden määrää ja näkyviin saatiin myös laitteiden verkkotunnukset. /12/

Käyttämällä sm-käskyä haluttiin varmistaa, että ilman Stargazer-ohjelmistoakin, saadaan seurattua laitteiden liikkumista verkossa, sekä saadaan tieto mikäli jokin laite menettää yhteytensä verkkoon käskyjä suoritettaessa. Laitteiden tiloja on havainnollistettu kuvioissa 18-19. /12/

```
> sm
     MAC               MoteId  State Nbrs Links Joins    Age StateTime
00-17-0D-00-00-60-3C-7E    1     Oper    0    19     1      0    0-00:21:49
00-17-0D-00-00-58-5E-2C    2     Lost    1     9     2    861    0-00:13:21

Number of motes (max 101): Total 2, Live 1, Joining 0
```

Kuvio 18. Managerin muodostaman verkon laitteiden tila, mote virta kiinni

```
> sm
     MAC               MoteId  State Nbrs Links Joins    Age StateTime
00-17-0D-00-00-60-3C-7E    1     Oper    1    21     1      0    0-00:29:44
00-17-0D-00-00-58-5E-2C    2     Oper    1     9     3     27    0-00:01:41

Number of motes (max 101): Total 2, Live 2, Joining 0
```

Kuvio 19. Managerin muodostaman laitteiden tila, mote virta päällä

Tiedonsiirron varmistamiseksi manager- ja mote-laitteen välillä suoritettiin käsky ”ping”, joka toimii lähettämällä vastausviestipyyntö laitteelta laitteelle. Vastausviestiksi motelta saatiin moten numero, aika, laitteen käyttämä virta ja lämpötila (**Kuvio 20.**). /10/

```
> ping 2
Sending ping request to mote 2

> Ping response from mote 2, time=1952 msec v=3093 t=24
```

Kuvio 20. Managerilta yhteystestaus moteen

Mikäli mote jostain syystä on kadonnut verkosta kesken kommunikoinnin, esimerkiksi sammunut tai hajonnut, manageri saa kuvion 21 mukaisen ilmoituksen.

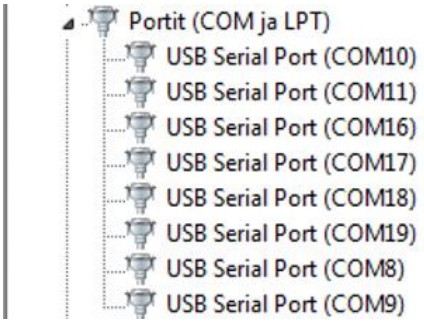
```
> PF: n=2 0 t=102714105 lh=100741495 d=1972610
3134593 : Path Alarm #1-#2
3134594 : MNGR_ERR_MOTELOST Path alarm for last path. Mote #2 is lost
3134633 : MNGR_ERR_MOTELOST Mote #2 change state to lost. Reason: PATHALARM
```

Kuvio 21. Managerin kommunikaatio motelle keskeytynyt

6.3.2 Mote DC9018A-B

Moten testeissä käytettiin yhtä motea porttien määrän takia. Tietokoneeseen mote liitettiin testilaitteiston DC9021A Interface/Dev Boardiin ja tämän kautta tietoko-

neeseen. Motet ovat paristokäyttöisiä, mutta Interface Boardin kautta mote voi ottaa virtansa USB-kaapelin välityksellä. Yhdistettäessä mote PC:hen, laite muodostaa 4 virtuaalista sarjaporttia, kuten manageria käytettäessä. Virtuaalisia sarjaportteja muodostuu siis 8 kun tietokoneeseen on liitettynä manager ja mote.



Kuvio 22. Managerin ja moten luomat virtuaaliset sarjaportit

Moteen yhdistäminen PuTTYllä tapahtuu samalla tavalla kuin manageria käytettäessä, konfiguraatioksi COM – portti, 9600 baud, 8 Data bits, Stop bits 1, parity none ja flow control none. Moten CLI-puolella ei testien kannalta löytynyt merkittäviä komentoja kokeiltavaksi, joten testeissä siirryttiin kokeilemaan testilaitteiston API-puolen käyttöä. /14/

6.4 Testilaitteiston API-puolen testaus

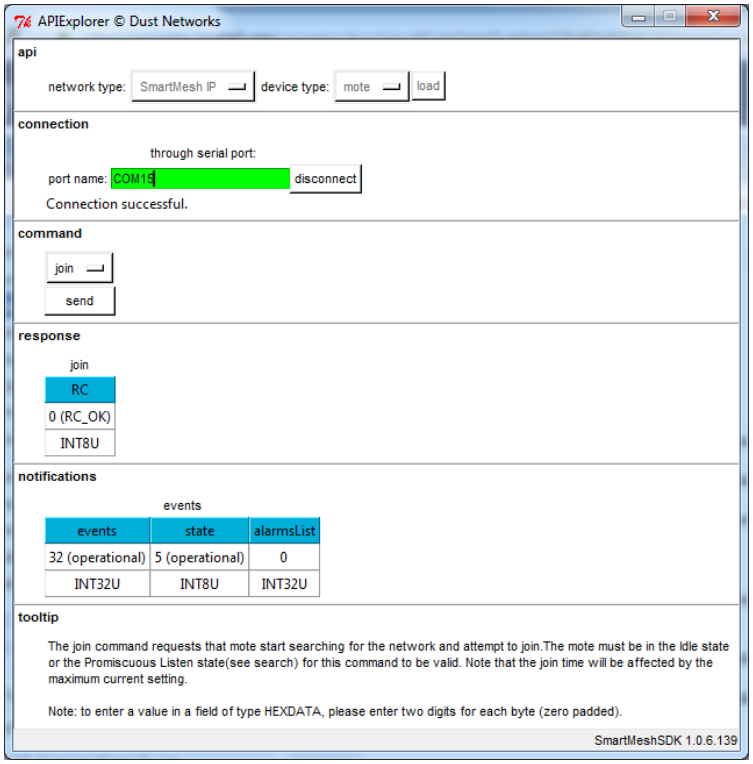
API puolen testauksessa keskityttiin kokeilemaan managerin ja moten välistä kommunikaatiota ja mahdollisia tarvittavia käskyjä laitteen toteuttamiseksi. Tärkeitä käskyjä laitteen toiminnan kannalta voidaan pitää verkkoon liittyminen(join), portin avaus kommunikoinnille (open socket), portin liittäminen avattuun kommunikointipisteeseen (bindSocket) ja viestin lähetys motelta managerille (sendTo). Apuna API-puolen testaukselle Linearilla on tarjolla SmartMesh SDK-ohjelmistopaketti, josta käytettiin komentojen testaukseen APIExplorer-ohjelmistoa. /13/

6.4.1 Esivalmistelut

Testeissä tietokoneeseen liitettiin manageri ja yksi mote. Manageri pidettiin vakiotilassa, jolloin se toimii master moodissa yrittäen muodostaa verkkoa. Motella mahdollisuuksina olivat master- (isäntä) tai slave (orja)-moodi, joista moodiksi valittiin slave-moodi, joka mahdollistaa moten API-puolen käytön. Master - slave asetukset asetettiin moten CLI-puolelta käyttämällä käskyä ”get mode”, jolla saatiin moten nykyinen moodi ja set mode slave, jolla moten tila vaihdettiin slaveksi. Mote käyttäytyy master moodissa siten, että se suorittaa siihen valmiiksi integroitua ohjelmaa, joka hallitsee verkkoon liittymistä ja generoi dataa. Slave moodissa mote odottaa sarjaliikenteellä yhdistettyä laitetta suorittamaan käskyjä ja hallitsemaan verkkoon liittymistä.

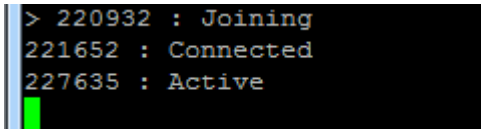
6.4.2 API Explorer yhden linkin toiminnalliset testit

APIExploreria käytettäessä managerin puolelta visualisoitiin liikennettä käyttämällä Stargazer ja PuTTY -ohjelmistoa. Ensimmäiseksi kokeiltiin moten liittymistä verkkoon suorittamalla ”join”-käsky (**Kuvio 23.**).



Kuvio 23. API Explorer ”join”-käsky

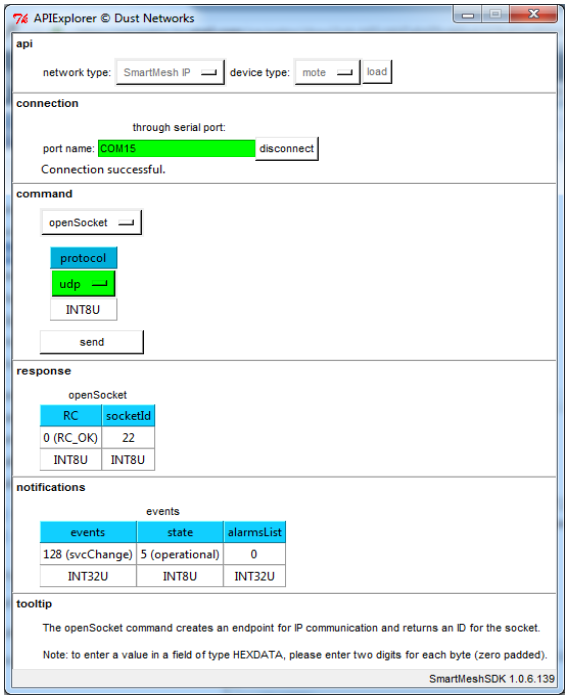
Ohjelmistosta valittiin virtuaalinen sarjaportti moten API-puolelle ja yhdistettiin siihen. Tämän jälkeen ohjelmiston komentovalikosta valittiin ”join”-käsky ja painettiin send, jonka jälkeen mote alkoi hakemaan verkkoa ja liittymään siihen. Mitään parametrejä ohjelmaan ei tarvinnut lisätä. Vastaukseksi onnistuneesta liittymisestä motelta tulee vastaus ”RC_OK”. Tiedot tapahtumasta moten liittymisestä verkkoon monitoroitiin selkeästi PuTTYllä (**Kuvio 24.**). /13/



Kuvio 24. Join PuTTYllä visualisoituna

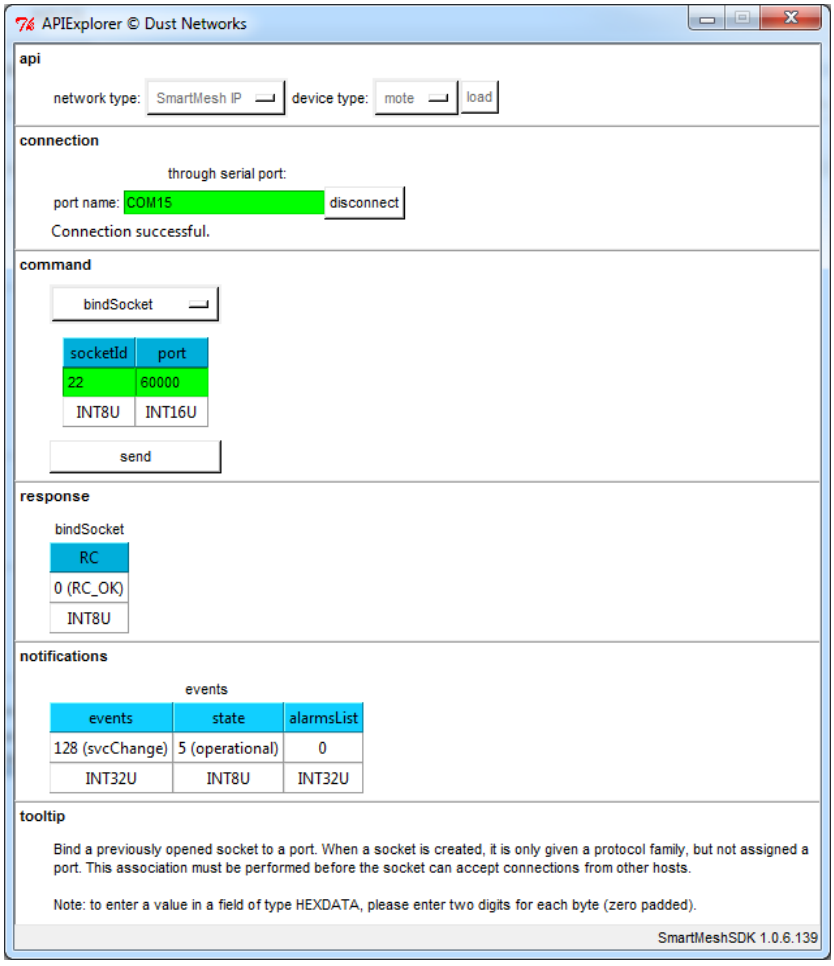
Kun käsky oli varmistettu toimivaksi, seuraavaksi kokeiltiin päätepisteen avausta IP-kommunikoinnille, joka tapahtui käskyllä ”openSocket”. Protokollana laitteistossa vaihtoehtona on UDP, joten sitä käytettiin testien yhteydessä. Vastaukseksi onnistuneesta päätepisteen avauksesta mote vastaa ”RC_OK” ja antaa numeron

päätepisteelle. Vakioasetuksilla moten antama päätepisteen numero alkaa 22:sta ja mikäli haluaa avata useamman päätepisteen, niin seuraavat ovat 22:sta suurempia numeroita. Visualisoinnissa tässä tapauksessa luotettiin APIExplorerin antamiin vastauksiin. Käskyn suorittaminen kuviossa 25. /13/



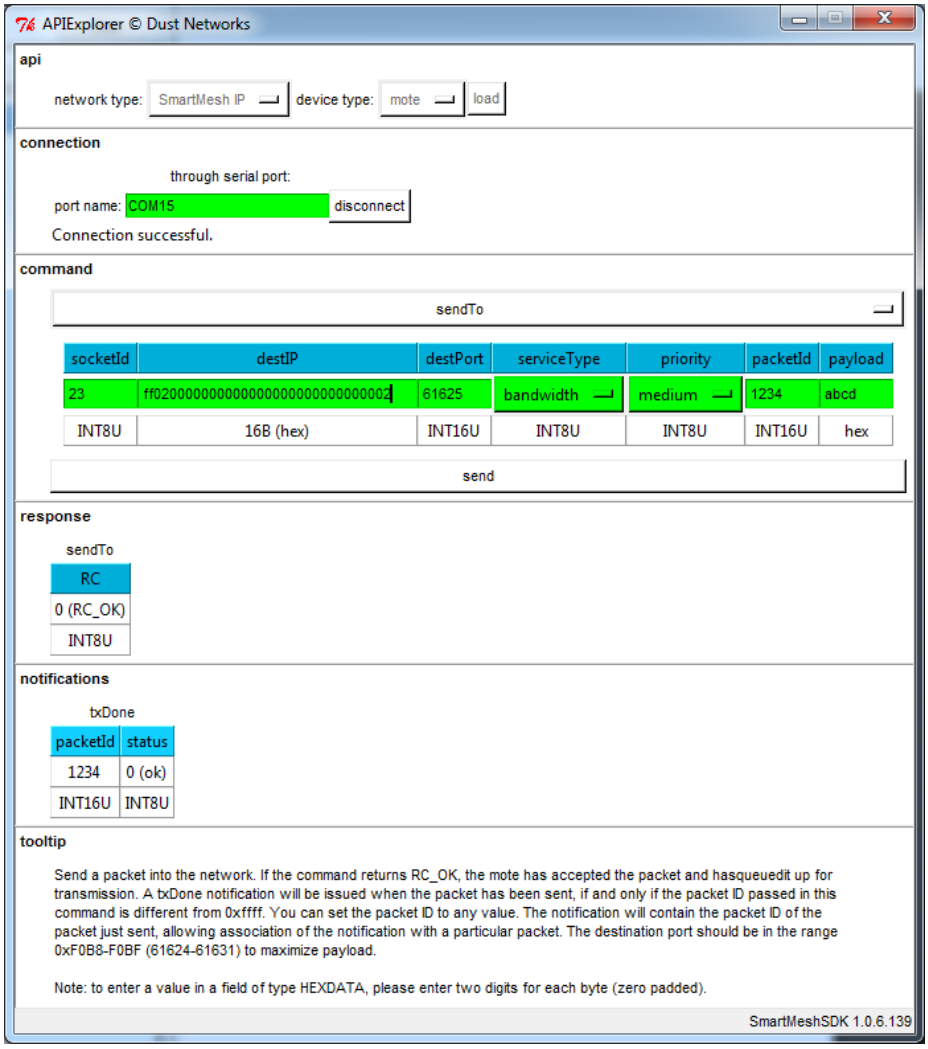
Kuvio 25. Open Socket-toiminnon onnistunut suorittaminen

Seuraavan komennon suorittamiseen vaaditaan ”openSocket”-käskyn antama numero päätepisteelle ja porttinumero, joka dokumentoinnin perusteella tulisi olla numeroiden 61624 – 61631 välillä maksimikapasiteetin saavuttamiseksi. Tämän käskyn visualisoimiseksi käytettiin APIExploreria, kuten edellä (**Kuvio 26.**). Vastaukseksi ohjelma ilmoittaa onnistuneesta käskyn suorittamisesta ”RC_OK”. /13/



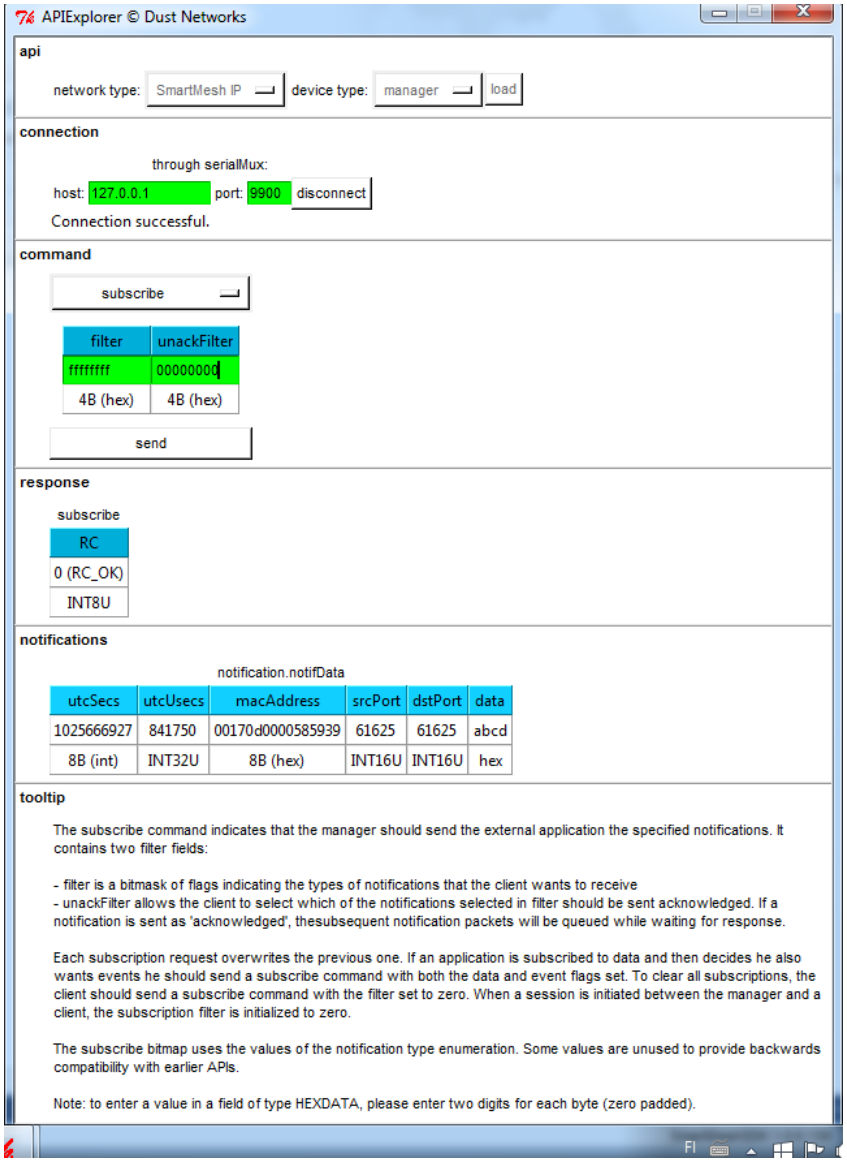
Kuvio 26. bindSocket-käskyn suorittaminen

Viestiliikenteelle avatun väylän jälkeen kokeiltiin viestin lähettämistä motelta managerille. Moten käskyistä tähän sopiva vaihtoehto oli ”sendTo”-toiminto, joka vaatii syötettäviksi parametreiksi päätepisteen numeron, vastaanottajan IP:n, vastaanottajan portin numeron, paketin numeron ja viestin. Onnistuneesta viestin lähettämisestä tulee motelta vastaus ”RC_OK” (**Kuvio 27.**). /13/



Kuvio 27. sendTo-toiminnon suorittaminen

Ennen viestin lähettämistä managerille täytyy avata subscribetila, jotta viesti saataisiin näkyviin APIExplorerilla. Tällä komennolla manageri ilmoittaa heti, jos jonkinlainen tapahtuma, esimerkiksi viesti on saapunut joltakin laitteelta managerille (**Kuvio 28.**). /13/



Kuvio 28. Managerin asettaminen subscribetilaan

Viestin lähetys todettiin onnistuneeksi APIExplorerissa ja tämä visualisoitiin vielä käyttäen Stargazer-ohjelmiston viestiliikenteen visualisoimiseen tehtyä ominaisuutta Traffic Monitor (**Kuvio 29.**).

Time	Direction	Mote	Length	Payload
7.7.2016 16:49:46	RX	59-39	28	14 04 00 00 00 00 3D 22 6F 6F 00 0C D8 16 00 17 0D 00 00 58 59 39 F0 B9 F0 B9 AB CD

Kuvio 29. Viestin kaappaus, StarGazer-liikenteen visualisointi

Näillä APIExplorerilla tehdyillä testeillä todettiin, että tarvittavat käskyt voidaan suorittaa testilaitteistolla ja voitaisiin lähteä rakentamaan omaa Windows-käyttöjärjestelmälle tarkoitettua ohjelmistoa, jolla ohjattaisiin laitteita.

6.5 Windows API Mote

Tässä opinnäytetyön osiossa tehtiin Windows-pohjainen ohjainohjelma Linear Dustin testilaitteistolle, jolla saataisiin välitettyä viestit tietyin välein motelta managerille. Ohjelma tehtiin C – kielellä, ja ohjelmiston kehitysympäristönä toimi Visual Studio 2012.

Tarkoituksena ohjelmalla toiminnaltaan oli muodostaa yhteys managerin muodostamaan verkkoon, avata päätepiste kommunikoinnille, määrittää avatulle päätepisteelle käytettävä portti ja lähettää viesti motelta managerille.

Ohjelman ensimmäinen vaihe oli luoda toimiva sarjaliikenneyhteys moten API-puolen kanssa. Tähän tarkoitukseen käytettiin apuna CreateFile()-funktiota, jolla voidaan luoda tai avata tiedosto tai I/O-laite (**Kuvio 30**).

```
hSerial = CreateFile("\\\\.\\COM15", //address serial
          GENERIC_READ|GENERIC_WRITE, //access (read - write) mode
          0, //share mode
          NULL,// address security description
          OPEN_EXISTING, // How to create
          FILE_ATTRIBUTE_NORMAL, // file attributes
          NULL // handle of file with attribute to copy
          );
```

Kuvio 30. CreateFile()-funktio

Tämän jälkeen varmistettiin yhteyden käyttämät parametrit SetCommState()- ja GetCommState-funktiota käyttäen. GetCommState()-funktiolla haetaan yhteyden

nykyiset määritykset ja SetCommState()-funktiolla asetetaan yhteyden uudet asetukset voimaan (**Kuvio 31.**).

```
Status = GetCommState(hSerial, &dcbSerialParams);//get current settings

fprintf(stderr, "Current Settings: \n Baud Rate: %d \n Byte Size: %d \n Stop bits: %d \n Parity: %d \n",
dcbSerialParams.BaudRate,dcbSerialParams.ByteSize,dcbSerialParams.StopBits,dcbSerialParams.Parity);

dcbSerialParams.BaudRate = CBR_115200;
dcbSerialParams.ByteSize = 8;
dcbSerialParams.StopBits = ONESTOPBIT;
dcbSerialParams.Parity   = NOPARITY;

SetCommState(hSerial, &dcbSerialParams);//set COM parameters
fprintf(stderr, "New Settings: \n Baud Rate: %d \n Byte Size: %d \n Stop bits: %d \n Parity: %d \n",
dcbSerialParams.BaudRate,dcbSerialParams.ByteSize,dcbSerialParams.StopBits,dcbSerialParams.Parity);
```

Kuvio 31. SetCommState() – ja GetCommState() – funktio

Sarjaliikenneyhteyden muodostuksen jälkeen voitiin aloittaa laitteella suoritettaviin käskyihin tarvittavien komentojen kasaaminen. Laitteen käskyjen muotona toimivat HDLC-kehyksen muotoiset käskyt ja ohjelmassa nämä käskyt kovakoodattiin parametri kerrallaan taulukoihin. Verkon etsimiseen ja liittymiseen tarvittava komento ”kovakoodattuna” kuviossa 32.

```
bytes_to_send[0] = 0x7E;
bytes_to_send[1] = 0x06;//cmd
bytes_to_send[2] = 0x00;//length
bytes_to_send[3] = 0x00;//Request
bytes_to_send[4] = 0x15;//FCS
bytes_to_send[5] = 0x10;//FCS
bytes_to_send[6] = 0x7E;
```

Kuvio 32. Join-käsky kovakoodattuna

Ohjelmassa muodostettiin samankaltaiset rakenteet käskyille opensocket, bindsocket ja sendTo. Järjestys, jossa käskyt haluttiin suorittaa olivat laitteen verkkoon liittyminen (join), luodaan päätepiste kommunikoinnille (opensocket), liitetään päätepiste porttinumeroon (bindsocket) ja lähetetään viesti laitteelta keskusyksikölle (sendTo).

Kun edelliset vaiheet oli tehty, seuraavaksi muodostettiin komennoille lähetykseen tarvittava koodi ja samalla viestien vastaanottamiseen tarvittava koodi. Viestien lähetysfunktiona ohjelmassa käytettiin WriteFile()-funktiota ja vastaanottami-

seen ReadFile()-funktiota. WriteFile()-funktiota käytettäessä yritettiin aluksi lähettää kasatut viestit kerrallaan laitteelle, mutta muutaman yrityksen jälkeen todettiin, että laite ei hyväksynyt viestejä tällä tavoin lähetettynä. Ongelman ratkaisemiseksi kokeiltiin lähetystä parametri kerrallaan ja todettiin, että laite ymmärsi käskyt parametri kerrallaan lähetettynä (**Kuvio 33.**).

```
fprintf(stderr, "Send bytes \n");
for(i = 0; i<=8; i++)
{
    Status = WriteFile(hSerial,&bytes_to_send[i],1, &written,NULL);
    fprintf(stderr, "%x \n", bytes_to_send[i]);
}
fprintf(stderr, "%d bytes written\n", written);
```

Kuvio 33. WriteFile()-funktio

Käskyn suorittamisen jälkeen luettiin vastaanotettava liikenne ReadFile()-funktiota apuna käyttäen (**Kuvio 34.**). Ongelmana lukemisessa oli, että vastaanotettavaa dataa tuli niin paljon, että oikean vastausviestin sisältävän osion rajaamista ei onnistuttu toteuttamaan.

```
do
{
ReadFile(hSerial, &TempBuffer, sizeof(TempBuffer), &NoBytesToRead, NULL);
SerialBuf3[i] = TempBuffer;

i++;
}while (NoBytesToRead > 0);
```

Kuvio 34. ReadFile()-funktio

Ohjelmassa ilmenneen vastaanotettavien viestien rajaamiseen liittyvästä ongelmasta huolimatta, saatiin aikaiseksi toimiva käskyjen suorittaminen laitteistolla, joten ohjelman kokoamisessa aloitettiin peräkkäisten viestien suorittamiseen tarvittavan toiminnollisuuden muodostaminen. Käskyjen haluttu suoritusjärjestys oli laitteen verkkoon liittyminen, päätepisteen luonti kommunikaatiolle kahden laitteen välillä, päätepisteen sitominen porttinumeroon ja viestin lähetys laitteelta laitteelle.

Halutut toiminnot kokeiltiin suorittaa aluksi käsky kerrallaan, jotta saataisiin varmistettua niiden hyväksyttävä rakenne. Kun toimintojen varmistaminen oli saatu toteutettua, alettiin kokeilemaan käskyjen peräkkäistä suorittamista, jonka myötä ilmeni ongelma, jossa peräkkäin lähetetyistä viesteistä jälkimmäisen sisältämä toiminnollisuus jäi laitteelta suorittamatta.

Ongelman ratkaisemiseksi kokeiltiin viiveen asettamista lähetettävien käskyjen väliin, jotta laitteisto ehtisi suorittamaan edellisen viestin sisältämän toiminnollisuuden. Ratkaisua ei löytynyt, joten seuraavaksi kokeiltiin tutkia onko viestien rakenteessa, jotakin vikaa. RFC1662-standardista löytyi maininta ”Only one Flag Sequence is required between two frames”, eli tarvittaisiin vain yksi lippusekvenssi kahden kehyksen välillä. Ohjelmassa yritettiin siis ratkaista ongelma poistamalla paketin välissä oleva lippusekvenssit 0x7E. Ongelma ei ratkennut, eikä löydetty muita mahdollisia ratkaisua ohjelman ongelman ratkaisemiseen, joten ongelmaa yritettiin ratkoa yhdessä laitteiston toimittajan kanssa. Ongelmaan ei saatu kehitettyä ratkaisua, joten ohjelmiston tekeminen jätettiin tähän ongelmakohtaan.

7 YHTEENVETO

Opinnäytetyön lopputuloksena saatiin lähes valmis ohjelmisto avojohtolinjan vaihevirtojen mittausjärjestelmän langattoman tiedonsiirron toteuttamiseksi. Testausosiosta saadaan tietoa ja näkemystä Linear Dust Networks valmistamista testilaitteistojen mahdollisuuksista ja niiden käyttämistä tekniikoista. Teoriaosuutta voidaan käyttää tehtäessä mahdollisia omia ratkaisuja vastaavaan käyttötarkoitukseen.

Ongelmat opinnäytetyössäni sijoittuivat ohjelmointiosioon opinnäytetyön lopussa. Monen peräkkäisen käskyn suorittamiseen liittyvien ongelmien takia, ohjelmistoa ei saatu toteutettua täysin valmiiksi. Tavoitteisiin ei täysin päästy ongelmista johtuen.

Jatkokehityksenä laitteistoon pitäisi lisätä sensoripuoli vaihevirtojen mittaukseen, sekä ratkaista kommunikaatioon liittyvä ongelma tai toteuttaa ohjelmisto jollakin toisella tavalla.

LÄHDELUETTELO

/1/ Bluetooth Class ratings
Viitattu 10.10.2017.
http://www.tomshardware.co.uk/forum/32906-39-bluetooth-class-ratings

/2/ Zigbee Specification
Viitattu 15.10.2017.
http://www.zigbee.org/download/standards-zigbee-specification/

/3/ Bluetooth, Zigbee and Wibree: A Comparison of WPAN technologies
Viitattu 15.10.2017.
https://cseweb.ucsd.edu/classes/fa08/cse237a/topicresearch/jkooker_tr_report.pdf

/4/ Mikä on LORAWAN?
Viitattu 16.10.2017.
https://digitamahdollistaa.fi/mika-on-lorawan/

/5/ Lora Alliance™ Technology
Viitattu 16.10.2017.
https://www.lora-alliance.org/technology

/6/ Hajaspektritekniikka
Viitattu 10.6.2017.
http://users.jyu.fi/~harsund/Arkisto/Sisalto.htm

/7/ TSMP: Time Sychronize Mesh Protocol
Viitattu 10.6.2017.
http://robotics.eecs.berkeley.edu/%7Epister/290Q/Papers/Pister%20TSMP%20DSN08.pdf

/8/ PPP in HDLC-like Framing
Viitattu 10.6.2017.
https://tools.ietf.org/html/rfc1662

/9/ SmartMesh IP User's Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/user-guide/SmartMesh_IP_User_s_Guide.pdf

/10/ SmartMesh IP Easy Start Guide for VManager
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/application-note/SmartMesh_IP_Easy_Start_Guide_for_VManager.pdf

/11/ SmartMesh IP Tools Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/software-and-simulation/SmartMesh_IP_Tools_Guide.pdf

/12/ SmartMesh IP Embedded Manager CLI Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/design-note/SmartMesh_IP_Embedded_Manager_CLI_Guide.pdf

/13/ SmartMesh IP Embedded Manager API Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/design-note/SmartMesh_IP_Embedded_Manager_API_Guide.pdf

/14/ SmartMesh IP Mote CLI Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/design-note/SmartMesh_IP_Mote_CLI_Guide.pdf

/15/ SmartMesh IP Mote API Guide
Viitattu 10.6.2017.
http://cds.linear.com/docs/en/design-note/SmartMesh_IP_Mote_Serial_API_Guide.pdf

/16/ Technical Overview of Time Synchronized Mesh Protocol
Viitattu 12.10.2017.
http://cds.linear.com/docs/en/white-paper/TSMP_Whitepaper.pdf

/17/ Core Version 5.0
Viitattu 10.10.2017.
https://www.bluetooth.com/specifications/bluetooth-core-specification

/18/ What is the Internet of Things (IoT)?
Viitattu 10.6.2017.
http://www.businessinsider.com/what-is-the-internet-of-things-definition-2016-8?r=US&IR=T&IR=T

/19/ ISM band
Viitattu 12.10.2017.
https://www.pcmag.com/encyclopedia/term/45467/ism-band

/20/ Classic Bluetooth vs Bluetooth low energy
Viitattu 5.10.2017.
http://www.mt-system.ru/sites/default/files/docs/documents/bluetooth_le_comparison.pdf

/21/ Another Bluetooth Low Energy 101 Primer
Viitattu 5.10.2017.
https://blog.adafruit.com/2016/10/25/another-bluetooth-low-energy-101-primer-ble-iot-iotuesday/

/22/ Granlund, K, 2007, Tietoliikenne, 1. painos maaliskuu 2007, Porvoo, WS Bookwell

/23/ Crypto Law Survey
Viitattu 10.11.2017.
http://www.cryptolaw.org/

/24/ Download PuTTY
Viitattu 10.11.2017.
http://www.putty.org/